Probabilistic retrieval models - relationships, context-specific application, selection and implementation

Wang, Jun

Probabilistic Retrieval Models - Relationships, Context-specific application, Selection and Implementation

Jun Wang

University of London

Thesis submitted for the degree of Doctor of Philosophy

at Queen Mary, University of London

December 2010

Declaration of originality

I hereby declare that this thesis, and the research to which it refers, are the product of my own work, and that any ideas or quotations from the work of other people, published or otherwise, are fully acknowledged in accordance with the standard referencing practices of the discipline.

The material contained in this thesis has not been submitted, either in whole or in part, for a degree or diploma or other qualification at the University of London or any other University. Some parts of this work have been previously published as:

- Jun Wang and Thomas Roelleke, "Statistical query features and correlation to performance of TF-IDF and language modelling" (Submitted)

- Thomas Roelleke, Jun Wang, Jan Frederick Forst and Hengzhi Wu, "Informativeness and Probability Mixtures: On the Symmetry of TF-IDF and Language Modelling" (In preparing for submission)

- Thomas Roelleke and Jun Wang, "TF-IDF Uncovered: A Study of Theories and Probabilities" in 31st ACM SIGIR Conference on Research and Development in Information Retrieval, Singapore, 2008

- Thomas Roelleke, Heng Zhi Wu, Jun Wang, and Hany Azzam. "Modelling retrieval models in a probabilistic relational algebra with a new operator: The relational bayes", VLDB Journal, 2008.

- Thomas Roelleke and Jun Wang, "Probabilistic Logical Modelling of the Binary Independence Retrieval Model" in 1st international conference on the theory of information retrieval, Budapest, 2007

- Thomas Roelleke and Jun Wang, "A Parallel Derivation of Probabilistic Information Retrieval Models" in 29th ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, 2006

- Jun Wang and Thomas Roelleke, “Context-specific Inverse Document Frequency for Structured Document Retrieval” in European Conference on Information Retrieval, London , 2006

jun wang

Abstract

Retrieval models are the core components of information retrieval systems, which guide the document and query representations, as well as the document ranking schemes. TF-IDF, binary independence retrieval (BIR) model and language modelling (LM) are three of the most influential contemporary models due to their stability and performance. The BIR model and LM have probabilistic theory as their basis, whereas TF-IDF is viewed as a heuristic model, whose theoretical justification always fascinates researchers.

This thesis firstly investigates the parallel derivation of BIR model, LM and Poisson model, wrt event spaces, relevance assumptions and ranking rationales. It establishes a bridge between the BIR model and LM, and derives TF-IDF from the probabilistic framework.

Then, the thesis presents the probabilistic logical modelling of the retrieval models. Various ways of how to estimate and aggregate probability, and alternative implementation to non-probabilistic operator are demonstrated. Typical models have been implemented.

The next contribution concerns the usage of of context-specific frequencies, i.e., the frequencies counted based on assorted element types or within different text scopes. The hypothesis is that they can help to rank the elements in structured document retrieval. The thesis applies context-specific frequencies on term weighting schemes in these models, and the outcome is a generalised retrieval model with regard to both element and document ranking.

The retrieval models behave differently on the same query set: for some queries, one model performs better, for other queries, another model is superior. Therefore, one idea to improve the overall performance of a retrieval system is to choose for each query the model that is likely to perform the best. This thesis proposes and empirically explores the model selection method according to the correlation of query feature and query performance, which contributes to the methodology of dynamically choosing a model.

In summary, this thesis contributes a study of probabilistic models and their relationships, the probabilistic logical modelling of retrieval models, the usage and effect of context-specific frequencies in models, and the selection of retrieval models.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to express my gratitude to my supervisor, Dr Thomas Roelleke for his advice, support and encouragement throughout the course of my Ph.D work.

Many thanks to everyone in QMIR group in Queen Mary, University of London for their help and support, especially Elham Ashoori and Zoltan Szlavik who shared the same office for many years and encouraged me as well as each other. And thanks to Professor Mounia Lalmas and Hany Azzam for helping with my writing, Tim Kay for his technical support and proof reading of this thesis.

Finally I would thank to my parents, who support me financially and spiritually. Without them I would not be able to start and finish my Ph.D thesis.

Notation of this thesis

Notation in this thesis	Description	Traditional notation
d	a document	
q	a query	
c	a collection	
$N_L(c)$	Number of locations in collection c	
$n_L(t,c)$	Number of locations at which term t occurs in collection c	
$N_D(c)$	Number of documents in collection c	N
$n_D(t,c)$	Number of documents in which term t occurs in collection c	df
$N_D(c)$	Number of documents in collection c	N
$n_D(t,c)$	Number of documents in which term t occurs in collection c	n
$N_D(r)$	Number of relevant documents in collection c	R
$n_D(t,r)$	Number of relevant documents in which term t occurs in collection c	r
$N_L(d)$	Number of locations in document d	dl
$n_L(t,d)$	Number of locations at which term t occurs in document d	tf
$\lambda(t,x) = \frac{n_L(t,x)}{N_D(x)}$	Average frequency of term t in the set x of documents	
$avgtf(t,x) = \frac{n_L(t,x)}{n_D(t,x)}$	Average frequency of term t in the t document set	
$avgdl(x) = \frac{n_L(x)}{n_D(x)}$	Average document length in document set x	
$P_L(t\|d) = \frac{n_L(t,d)}{N_L(d)}$	Probability that term t occurs in a location of document d	
$P_L(t\|c) = \frac{n_L(t,c)}{N_L(c)}$	Probability that term t occurs in a location of collection c	
$P_D(t\|r) = \frac{n_D(t,r)}{N_D(r)}$	Probability that t occurs in a relevant document, also noted as $P_{BIR}(t\|r)$	p_t
$P_D(t\|\overline{r}) = \frac{n_D(t,\overline{r})}{N_D(\overline{r})}$	Probability that t occurs in a non-relevant document, also noted as $P_{BIR}(t\|\overline{r})$	q_t
$P_D(t\|c) = \frac{n_D(t,c)}{N_D(c)}$	Probability that t occurs in a document of collection c, also noted as $P_{BIR}(t\|c)$	
$RSV_{BIR}(d,q)$	BIR retrieval status value, $RSV_{BIR}(q,d) = O(r\|d) \stackrel{rank}{=} \sum_{t\in d\cap q} \log w(t)$, $w(t) = \frac{\frac{n_D(t,r)}{N_D(r)} \cdot \frac{N_D(c)-N_D(r)-(n_D(t,c)-n_D(t,r))}{N_D(c)-N_D(r)}}{\frac{n_D(t,c)-n_D(t,r)}{N_D(c)-N_D(r)} \cdot \frac{N_D(r)-n_D(t,r)}{N_D(r)}}$	
$RSV_{LM}(d,q)$	LM retrieval status value , $RSV_{LM}(d,q) = P_{LM}(q\|d) = C \cdot \sum_{t\in q\cap d} \log(1 + \frac{\lambda P(t\|d)}{(1-\lambda)P(t\|c)})$	
$RSV_{PM}(d,q)$	Poisson retrieval status value, $RSV_{PM}(d,q) = O(r\|d) = \frac{P(d\|r)}{P(d\|\overline{r})} = \prod_{t\in d\cap q} \frac{P_{Poisson}(t\|r)}{P_{Poisson}(t\|\overline{r})} \cdot \prod_{t\in q\setminus d} \frac{P_{Poisson}(t\|r)}{P_{Poisson}(t\|\overline{r})}$	
(T,P)	Probabilistic relation, which consists of a tuple set T and a probability set P. T and P have the same set size, each probability value corresponds a tuple.	
τ	Tuple in a relation, which contains a few attributes $(\tau[1], \tau[2] ... \tau[n])$.	
i_n	Index of an attribute in a tuple. Therefore $\tau[i_1, i_2, ... i_n]$ is a new tuple which contains of the attributes $i_1, i_2, ... i_n$ from tuple τ.	
$P(\tau)$	Probability of tuple τ.	

Table 1: Notation overview

Chapter 1

Introduction

Information retrieval (IR) is a field of science concerned with searching collections for the objects relating to a searcher's interests, which can be text, image, sound, etc. Text retrieval was initially developed in the 70's, and was concerned with keyword or abstract retrieval. Today it has been developed into full document retrieval in a very large-scale. The collections concerned can be of terabyte size, centralized or distributed on a network.

The main components of an IR system are document and query representations, retrieval models (or matching functions), and some systems include relevance feedback parts. Document and query representations are sets of indexed terms which are tokenized from documents and queries, each term is presented with its occurrence information. Retrieval models decide how to match the documents and queries and how to assign a score to a document with respect to a query. How to index document and query terms, and what kind of statistical information is to be represented along with the terms, are also decided by the retrieval models. The relevance feedback part allows a system to gather relevance information and use it in a further search. Not all the retrieval systems have a relevance feedback part.

The retrieval model is the key component of a retrieval system, where the model guides the representations of queries and documents, and the ways of matching between them. In past decades, there have been various models developed, mainly divided into three categories: Boolean models, vector space models, and probabilistic models. Even the probabilistic model includes many different models, such as the famous binary independence retrieval model, Poison model, language modelling, di-

vergence from randomness, inference network, and many others derived from them [Robertson and Sparck Jones, 1976, Ponte and Croft, 1998a, Roelleke and Wang, 2006, Amati and van Rijsbergen, 2002, Ponte and Croft, 1998a, Robertson and Walker, 1994, Robertson et al., 1995, Lavrenko and Croft, 2001, Hiemstra, 2000, Zaragoza et al., 2003, Zhai and Lafferty, 2004, Harter, 1975a, Harter, 1975b, Crestani et al., 1998] .

The research of this thesis started from probabilistic logical modelling introduced in chapter 4, which aims to bring to IR model implementation with flexible data management, and concise form, as well as the ranking ability on objects beyond text. During our implementing, following research questions inspired us to investigate the relationships of the models, context-specific frequencies and model selection:

- With the existing probabilistic models, some of them use the probabilities of terms appearing in the documents and the probabilities of terms appearing the collection. When they represent the probability of a term occurring in a document collection, they use $P(t|c)$. However, in different models, $P(t|c)$ are estimated in different ways. Some are defined as the number of documents where the the term appears divided by the total number of documents in the collection, others are defined as the number of times that the term appears in the collection divided by the total number of terms in the collection. So the questions are: What are the relationships between models? For those same naming probabilities from different probabilistic models, what are the relationships between them?

- There is a non-probabilistic model, TF-IDF[Salton et al., 1975], long been famed for its simple formulation and robust performance over different collections. TF-IDF model also uses the frequencies of terms appearing in the documents and the frequencies of terms appearing the collection to weight a term. IDF started from intuition, and it has been given an approximation to probabilistic model [Robertson, 2004]. In the past few years, TF-IDF has been investigate from probability and information theory by [Aizawa, 2003, Hiemstra, 2000, Roelleke, 2003a]. There remain the questions : what is the relationships of TF-IDF to those probabilistic models, could we give a further theoretical justification to the TF-IDF?

- When using inverse document frequency IDF[1] to weight how effective a term can distin-

[1]IDF, inverse document frequency, is the logarithms of the collection size divided by the number of documents containing the term. Please find detailed definition section 2.9.3

guish different documents, we found that IDFs computed within different document sets can be different. For example the term “Artificial”’s IDF for the sub-collection concerning about artificial intelligence, also different to the IDF for the sub-collection concerning computer graphics. How to choose an appropriate collection containing the term to get a proper IDF value is one question. When we replace the document in IDF definition with section or paragraph, we can have different IDF values even if they are compute within the same collection. Which type of IDF can achieve better retrieval result is also a question. How to apply these different frequencies to probabilistic models, and what is outcome of utilizing these frequencies will be worthwhile to investigate?

- When running different retrieval models on a set of queries, we found that no one retrieval model can outperform other against whole set of queries. Therefore, we expect if we can find for each query a suitable retrieval model that is likely to perform the best than other models, then we can improve the overall high retrieval quality over a set queries without improving the existing models or developing new models. The questions are whether we are able to choose the model that performs the best, and what is the criteria for the model selection?

- IR systems assign weights to terms according to the retrieval models, and instantiate term weight to facilitate the retrieval process. When a system incorporates more than one retrieval models, it needs to maintain different term weights tailed to corresponding models. All these weights are normally collection dependent, they need to be updated when the collection changes with time. If the term weights or related probabilities can be estimated during the retrieval time from basic occurrence information, such as term-document relation, then the retrieval system just need to maintain a copy of term occurrence information. Whenever new documents are included into the the collection, only the additional term occurrence information is to be appended, or when some documents are removed from the collection, only the related term occurrence information is to be removed. Hence, no extra work will be involved. The probabilistic relational language can meet such expectation, and provide great re-usability of data in modelling. However, the probabilistic modelling is not straightforward as not every operation in retrieval models are probabilistic, which require us to look for a alternative way to explain and implement them with the probabilistic relational language.

1.1 Contributions

This thesis investigates the following aspects of probabilistic models: relationships of the models, probabilistic modelling, application of context-specific frequencies in probabilistic models, model selection based on the correlation between statistical query features and retrieval performance.

1.1.1 Relationships between Retrieval Models

This thesis investigates the probabilistic models (Binary independence model, Poisson model and Language modelling) and TF-IDF. The strictly parallel investigations show: 1. The clarification of probability spaces, relevance assumptions, and ranking rationales for the probabilistic models are displayed in parallel. 2. The Poisson model can be viewed as a bridge that connects BIR model and LM (Poisson bridge). 3. Both the BIR and LM can be used in a dual way for representing TF-IDF. 4. Under disjointness and independence assumptions, the decompositions of relevance probability lead to TF-IDF justification with some extreme but meaningful assumptions. 5. TF-IDF is the integral of document and query independence model, which gives TF-IDF another justification.

1.1.2 Probabilistic Modelling of Retrieval Models

Probabilistic modelling enables integrating probabilistic estimation and evaluation into the modelling, which brings a great flexible data management, also the readability and re-usability of the code. Although there are difficulties in modelling non-probabilistic retrieval models and adaptions of the models are to be made, the adaption lead to variations of the retrieved models. Some of the adaptions actually simplify the operation during retrieval time, some bring better retrieval performance than the original ones.

1.1.3 Context-specific Frequencies in Retrieval Models

Although context is a confusing term in information retrieval, and applies to anything related to the retrieval process, here it is about the text collection surrounding a document or a part of a document. According to the different contexts, the frequencies involved can be section or paragraph frequencies in addition to document frequencies. The application of context-specific frequencies in the retrieval models, brings a general retrieval model, where a retrieval object can be any type of element in a document, and the frequencies are chosen based on the retrieval

object. It also brings to the retrieval systems great scalability and flexibility, and can also be extended into distributed retrieval.

1.1.4 Model Selection by Statistic Query Features

Model selection comes from the observation that different retrieval models may work differently on the same query. If it is possible to choose for each query an appropriate retrieval model from existing ones, then, without elaborating the retrieval models, overall better performance can be achieved. The last chapter of this thesis contributes an extensive study and methodology to exploit the query features and to identify the necessary conditions for improvements through model selection.

1.2 Overview of the Thesis

Chapter 2 introduces some retrieval models and related term weighting schemes.

Chapter 3 investigates the derivations of BIR model, PM and LM, and explains the TF-IDF model from different aspects: with Poisson bridges, the two main probabilistic model the binary independence retrieval model and language modelling can be used to represent TF-IDF in dual ways; starting from decomposition of relevance probability based on independent or disjoint terms, TF-IDF can be derived with some extreme assumptions; decomposition based on disjoint term leads to the document and query independence measurement that relates to TF-IDF.

Chapter 4 introduces how to use highly abstract languages to implement the retrieval models, and shows how to integrate probability estimation into modelling and how different probabilistic assumptions can affect the retrieval results. In this chapter, alternatives ways are presented to model the non-probabilistic model in the probabilistic framework.

Chapter 5 applies context-specific frequencies to IDF and language modelling to retrieve the element or document in structured document collection. Collection selection by context-specific frequencies is also studied. An alternative informativeness measurement to IDF is investigated in this chapter too.

Chapter 6 proposes a way to choose a suitable retrieval model according to correlation of statistical query features and query performance. The ranking correlation of the models and the correlation of statistical query features and query performance are detailed studied.

Chapter 7 summarizes the work that has been done so far and provides an overview of future

research.

Chapter 2

Background

A brief history of retrieval models is given in this chapter, and then some classical models based on set theory, probabilistic theory and algebra are introduced. At the end of this chapter, probabilistic estimation methods related to these models are described.

2.1 History of Retrieval Models

The earliest retrieval model is the Boolean model coming from database retrieval, which returns the documents satisfying a Boolean expression. The standard Boolean model does not rank the retrieved documents. In the early 80s, [Bookstein, 1980, Salton et al., 1983] started to use fuzzy logic or extended Boolean operators to support document ranking.

In the early 60s, [Maron and Kuhns, 1960] introduced probabilistic theory into IR. In the 70s, [Robertson and Sparck Jones, 1976] proposed the binary independence retrieval (BIR) model based on the probabilistic ranking principle [Robertson, 1977]. [Croft and Harper, 1979] extended the binary independence retrieval model to the retrieval corpus without relevance information. [Harter, 1975a, Harter, 1975b] investigated 2-Poisson model in term weighting aspect, and [Margulis, 1992] studied the N-Poisson model. Their works showed that the 2-Poisson model can best express term distributions. [Fuhr and Buckley, 1991] devised a feasible probabilistic indexing model, which is the counterpart of the BIR model. In the mid 90s, [Robertson and Walker, 1994, Robertson et al., 1995] simplified the 2-Poisson model, and created the BM25 model by combining 2-Poisson and BIR models, thus far BM25 is one of the best retrieval models. There were another two branches of probabilistic model lying from the 80s and

90s, [van Rijsbergen, 1986, van Rijsbergen, 1989] introduced non-classical probabilistic logic $P(d \rightarrow q)$ to IR models, [Wong and Yao, 1995] generalized the logic model $P(d \rightarrow q)$, which can express other models. [Turtle and Croft, 1990, Turtle and Croft, 1991] applied Bayesian inference networks to document retrieval. From the end of 90s, [Ponte and Croft, 1998a] first applied language modelling to IR, and a lot of work [Lavrenko and Croft, 2001, Hiemstra, 2000, Zaragoza et al., 2003, Zhai and Lafferty, 2004] based on that has been carried out since then. In the early 2000s, a probabilistic framework based on divergence from randomness appeared as another successful model, which utilizes different term distributions in the model, and can also represent some other existing probabilistic models [Amati and van Rijsbergen, 2002]. For more detailed information about probabilistic models, please refer to [Crestani et al., 1998], [Fuhr, 1992] and [Baeza-Yates and Ribeiro-Neto, 1999].

The last type of retrieval model is the vector space model, which started in the early 70s [Salton, 1971, Salton et al., 1975]. Originally term weights were binary, later terms are weighted by their occurrences within the document (TF) and inverse document frequencies (IDF, see section 2.9.3) due to the retrieval systems being able to index full documents. The weight with combination TF and IDF greatly improved retrieval performance. [Rocchio, 1971] included users interactive relevance feedback into the models. However, the vector space model was criticized for conflicting with the principle of vector space model in [Wong and Raghavan, 1984]. Because there is dependence between terms, it is inappropriate to treat those terms as orthogonal. [Wong et al., 1985, Raghavan and Wong, 1999] solved this problem through the use of general vector space model which projects the terms into n orthogonal dimensions first, then uses the new orthogonal vectors to represent the query and document. The latent semantic indexing model is another kind of model based on algebra which indexes the documents in a much lower dimension space [Dumais et al., 1988, Deerwester et al., 1990].

With the fast development of the world wide web in the 90s, [Brin and Page, 1998, Kleinberg, 1999] applied linkage information of webpages into retrieval models to improve web retrieval, PageRank and Hits are the two of the most effective web retrieval models.

2.2 Boolean Model

The Boolean model is based on set theory and Boolean logic. The documents are regarded as sets of terms. A document is relevant to a query if the document makes the query formula true. For

example, for a query "*a* AND *b*", terms "*a*" and "*b*" must be all contained in retrieved documents. For another query, "*a* OR *b*", either "*a*" or "*b*" being contained in the documents will make the query successful.

The Boolean model can be easily and rapidly calculated, and was widely used in early commercial retrieval systems. The drawbacks of Boolean model are the difficulty for inexperienced end users to formulate their information needs into strict defined Boolean expressions, no ranking of the retrieved documents, and either no or too many retrieval results sometimes.

There are some extended works on Boolean model, like the fuzzy logic [Bookstein, 1980] and the extended Boolean model [Salton et al., 1983], which address the ranking problem by redefining Boolean operator and giving a term an original weight based on TF and IDF.

2.3 Vector Space Model

The vector space model (VSM) treats both queries and documents as n-dimension vectors $\overrightarrow{q}(q_1, q_2 ... q_n)$ and $\overrightarrow{d}(d_1, d_2, ... d_n)$. The angle between the two vectors $\overrightarrow{q}$ and $\overrightarrow{d}$ acts as an estimation of the similarity or relevance between the document and the query: the bigger the angle is, the less similar the document and query are, and vice versa. Therefore the similarity score is given by the cosine value of the angle:

$$sim(d,q) = \frac{\overrightarrow{d} \cdot \overrightarrow{q}}{\sqrt{|\overrightarrow{d}|^2 \cdot |\overrightarrow{q}|^2}} \tag{2.1}$$

In the basic representation of VSM, the components of the vector are binary. In other words, if the term "t_i" occurs in the document, then the corresponding coordinate of "d_i" is 1, otherwise 0. Therefore, the basic representation will not take into account the term frequencies within the document. Salton and Buckley [Salton and Buckley, 1988] suggested using TF-IDF as the coordinate if a term occurs in a document, and this provides a much better retrieval quality than basic VSM.

TF-IDF model is a typical kind of vector space model, which assign term weight with $tf(t,d) \cdot idf(t)$ as a coordinator, where $tf(t,d)$ is number of times that a term appears in a document, and $idf(t)$ is the inverse document frequency of a term, i.e. the number of document s in the collection divided by the number of documents containing term t in the collection. Logarithms is applied to changed the $idf(t)$?s score scale. Detailed information about IDF will be discussed in the later section.

2.3.1 General Vector Space Model And Latent Semantic Analysis

The issue around VSM is that it assumes terms are independent to each other (orthogonal in the vector space), which is not the necessary case. A general vector space model (GVSM) was proposed in [Wong et al., 1985] which decomposes a term into m-dimension orthogonal concepts, $\overrightarrow{t} = G \cdot \overrightarrow{m}$. G is the representation matrix. Then the transformed representation $\overrightarrow{d'}$, $\overrightarrow{q'}$ and their similarity as follows.

$$\overrightarrow{d'} = \overrightarrow{d} \cdot G \tag{2.2}$$

$$\overrightarrow{q'} = \overrightarrow{q} \cdot G \tag{2.3}$$

$$sim(\overrightarrow{d'}, \overrightarrow{q'}) = \overrightarrow{d} \cdot GG^T \overrightarrow{q}^T \tag{2.4}$$

$$\tag{2.5}$$

Latent semantic analysis (LSA) can be viewed as a special case of GVSM. LSA is a technique in natural language processing, which analyzes relationships between a set of documents and the terms and generate a set of concepts related to the documents and terms [Deerwester et al., 1990]. Then the documents and queries are represented by the concepts identified by LSA. The projection of the document or query vector to lower dimensional space can remove the noise represented in the document, also can decrease the computational cost by removing the less important concept concept.

To carry on a LSA process, a term-document matrix has to be created first. Suppose there are m documents in the collection, and vocabulary size is n, the matrix will be:

$$\begin{matrix} & & & d & & \end{matrix} \tag{2.6}$$

$$A = \begin{pmatrix} t_{1,1} & t_{1,2} & t_{1,3} & \cdots & t_{1,m} \\ t_{2,1} & t_{2,2} & t_{2,3} & \cdots & t_{2,m} \\ t_{3,1} & t_{3,2} & t_{3,3} & \cdots & t_{3,m} \\ \vdots & & & \ddots & \vdots \\ t_{n,1} & t_{n,2} & t_{n,3} & \cdots & t_{n,m} \end{pmatrix} t \tag{2.7}$$

Here, $t_{i,j}$ is the term weight for term t_i in document d_j, which can be binary weight. If t_i appears in d_j, then $t_{i,j}$=1; otherwise $t_{i,j}$=0. $t_{i,j}$ can also be setup with other weighting schemes.

With singular vector decomposition (SVD), matrix A can be decomposed to the inner product of three matrix.

$$A = U \cdot S \cdot V^T \tag{2.8}$$

Where U is term-concept matrix, V is document-concept matrix, S is singular value matrix. U and V are orthogonal matrix. By reducing the rank of singular matrix S, the less important information or noise can be removed.

According to decomposition 2.8, the document transformation is $v[i] = d_i' = d_i \cdot U \cdot S^{-1}$. Then a query will be transformed in the same way $q' = q \cdot U \cdot S^{-1}$. Consequently, documents and queries can be compared in concept space.

2.4 Probabilistic Retrieval Model

Probabilistic retrieval model also refers to binary independence retrieval model, which tries to estimate the probability that a user will find a document relevant to a query, according to the term occurrences in the relevant document set and non-relevant document set. The idea behind probabilistic retrieval model is the probabilistic ranking principle. In this section we introduce the probabilistic ranking principle, binary independence retrieval model, and using the binary independence model with little or no relevance information.

2.4.1 Probabilistic Ranking Principle (PRP)

Maron and Kuhns [Maron and Kuhns, 1960] suggested that: since the retrieval system needs to deal with uncertainty, it is better to use probability, and give users the retrieval result according to the probability of relevance. If it is difficult for the system to give a probability value, then the alternative is the ranking of the probability. This is the beginning of the probabilistic Ranking principle, and Robertson formalized the PRP in [Robertson, 1977]:

> If a reference retrieval system's response to each request is a ranking of the documents in the collection in order of decreasing probability of usefulness to the user who submitted the request where the probabilities are estimated as accurately a possible on the basis of whatever data made available to the system for this purpose, then the overall effectiveness of the system to its users will be the best that is obtainable on the basis of that data.

Robertson justified the PRP optimal retrieval procedure though decision theory in [Robertson, 1977]. If the cost of reading a non-relevant document is a_1 and the cost of missing a relevant document is a_2, then the minimal cost of retrieving a document d will be:

$$a_2 \cdot P(r|d) > a_1 \cdot P(\bar{r}|d) \tag{2.9}$$

or:

$$a_2 \cdot P(r|d) > a_1 \cdot (1 - P(r|d)) \tag{2.10}$$

then:

$$p(r|d) > \frac{a_1}{a_1 + a_2} \tag{2.11}$$

Thus documents can be ranked by the relevance probability , and $\frac{a_1}{a_1+a_2}$ is the cut-off of the probability ranking list.

2.4.2 Binary Independence Retrieval (BIR) Model

The BIR model ranks a document according to the probability that users think the document relevant to a query $P(r|d)$. Due to the difficulties of estimating the relevance probability, the BIR model estimates the probability that a document occurs in the relevant document set via Bayes theorem, instead of the probability of a document being relevant.

$$P(r|d) = \frac{P(d|r) \cdot P(r)}{P(d)} \tag{2.12}$$

Using odds $O(r|d)$ instead of $P(r|d)$ will remove $P(r)$ and $P(d)$ from the probability estimation.

$$O(r|d) = \frac{P(r|d)}{P(\bar{r}|d)} = \frac{P(d|r)}{P(d|\bar{r})} \tag{2.13}$$

Under term independence assumption, the probability of a document being relevant can be estimated by the terms' occurrence probability in the relevant and non-relevant set:

$$P(d|r) = \prod_{t \in d} P(t|r) \tag{2.14}$$

Independence assumption means that terms appearing in the document are considered independently from each other, i.e. there is no association between terms. This assumption is not true in reality, but in practice it simplifies the modelling work and gives acceptable retrieval quality.

When it is assumed that only the query terms play a role in ranking, in other words, non-query terms have the same probability to occur in relevant and non-relevant documents, $O(r|d)$ will be written as:

$$O(r|d) = \frac{P(d|r)}{P(d|\overline{r})} = \frac{\prod_{t\in d\cap q} P(t|r) \times \prod_{t\in q\setminus d} P(\overline{t}|r)}{\prod_{t\in d\cap q} P(t|\overline{r}) \times \prod_{t\in q\setminus d} P(\overline{t}|\overline{r})} \tag{2.15}$$

When equation 2.15 is multiplied by $1 = \prod_{t\in d\cap q} \frac{P(\overline{t}|\overline{r})\cdot P(\overline{t}|r)}{P(t|\overline{r})\cdot P(t|r)}$, then the $O(r|d)$ will be:

$$\begin{aligned} O(r|d) &= \prod_{t\in d\cap q} \frac{P(t|r)P(\overline{t}|\overline{r})}{P(t|\overline{r})P(\overline{t}|r)} + \prod_{t\in q} \frac{P(\overline{t}|r)}{P(\overline{t}|\overline{r})} &(2.16)\\ &\propto \prod_{t\in d\cap q} \frac{P(t|r)P(\overline{t}|\overline{r})}{P(t|\overline{r})P(\overline{t}|r)} &(2.17)\end{aligned}$$

The second of part of equation 2.16, $\prod_{t\in q} \frac{P(\overline{t}|r)}{P(\overline{t}|\overline{r})}$, can be removed as it is estimated for the whole query and is the same for any document. The first part of equation 2.16 is the product of term weights for all the common terms between the document and the query. The weight for the query term is:

$$w(t) = \frac{P(t|r)P(\overline{t}|\overline{r})}{P(t|\overline{r})P(\overline{t}|r)} \tag{2.18}$$

The estimation of term occurrence probability in the relevant or non-relevant documents are based on document frequency, and they are listed in equation 2.19:

$$\begin{aligned} P(t|r) &= \frac{n_D(t,r)}{N_D(r)},\ \ P(\overline{t}|r) = 1 - P(t_i|r) &(2.19)\\ P(t|\overline{r}) &= \frac{n_D(t,c) - n_D(t,r)}{N_D(c) - N_D(r)},\ \ P(\overline{t}|\overline{r}) = 1 - P(t|\overline{r}) \end{aligned}$$

By replacing equation 2.19 into equation 2.18, the probabilistic term weight will be:

$$w(t) = \frac{\frac{n_D(t,r)}{N_D(r)} \cdot \frac{N_D(c) - N_D(r) - (n_D(t,c) - n_D(t,r))}{N_D(c) - N_D(r)}}{\frac{n_D(t,c) - n_D(t,r)}{N_D(c) - N_D(r)} \cdot \frac{N_D(r) - n_D(t,r)}{N_D(r)}} \tag{2.20}$$

Notation:

$P(t|r)$: the probability that term t occurs in relevant documents.

$P(t|\overline{r}$: the probability that term t occurs in non-relevant documents.

$P(\overline{t}|r)$: the probability that term t does not occur in relevant documents, which equals $1 - P(t_i|r)$.

$P(\overline{t}|\overline{r})$: the probability that term t does not occur in non-relevant documents, which equals $1-P(\overline{t_i}|r)$.

d,q : $d=\{t_1,t_2,...t_n\}$, d is a document consisting of terms; $q=\{t_1,t_2,...t_m\}$, q is a query with query terms.

$d\cap q$: all terms that occur in both the document d and the query q.

$q\setminus d$: all terms that occur in the query q, but not in the document d.

$w(t)$: term weight for term t, which is termed BIR term weight.

$n_D(t,c)$: the number of documents in which term t occurs.

$N_D(c)$: total number of documents in the collection c.

$n_D(t,r)$: the number of relevant documents in which term t occurs.

$N_D(r)$: total number of relevant documents.

As log-odds keeps the ranking of $P(r|d)$, the document's BIR retrieval status value (RSV), the score assigned by retrieval model with respect to the query,is defined as equation 2.21:

$$RSV_{BIR}(q,d)=O(r|d)\overset{rank}{=}\sum_{t\in d\cap q}\log w(t) \tag{2.21}$$

The logarithms in the retrieval status value formula is to make the term weight addable. As it is a monotonic function, it will keep the ranking the of odds, thus it will not change the document's probability ranking.

For better understanding of term weight estimation in BIR model, a simple example follows: Run a query "computer" on a collection c with 1000 documents ($N_D(c)=1000$), in which 20 documents are relevant to the query ($N_D(r)=20$). In these 20 relevant documents, only 15 documents contain the term "computer" ($n_D(t,r)=15$); while in the whole collection, there are 40 documents containing the query term "computer" ($n_D(t,c)=40$).

The probabilities and term weight estimation are the formula 2.22 and 2.23:

$$p(t|r)=\frac{n_D(t,r)}{N_D(r)}=\frac{15}{20}=\frac{3}{4},\quad p(t|\overline{r})=\frac{n_D(t,c)-n_D(t,r)}{N_D(c)-N_D(r)}=\frac{40-15}{1000-20}=\frac{5}{196} \tag{2.22}$$

$$w(t)=\log\frac{\frac{n_D(t,r)}{N_D(r)}\cdot\frac{N_D(D)-N_D(r)-(n_D(t,C)-n_D(t,r))}{N_D(C)-N_D(r)}}{\frac{n_D(t,C)-n_D(t,r)}{N_D(C)-N_D(r)}\cdot\frac{N_D(r)-n_D(t,r)}{N_D(r)}}=\log\frac{\frac{15}{20}\cdot\frac{955}{980}}{\frac{25}{980}\cdot\frac{5}{20}}=\log\frac{573}{5} \tag{2.23}$$

The times of a term occurring in a document (term frequency, TF) is involved in the BIR term weighting scheme. For the early abstract or keyword indexing systems, where there were not many repeated words in a abstract, thus TF was not so important. However TF plays a different role in full document retrieval systems, given the phenomenon that some terms appear in a document many more times than the others when the document is about a certain subject. Including 2-Poisson estimated TF and document length in the retrieval function together with BIR term weight leads to best match function (BM25), which will be introduced in section 2.6

2.4.3 Variations of the BIR Term Weight

BIR term weighting takes into account relevance and non-relevance, as well as occurrence and non-occurrence information. When there is not enough such information involved during the estimation, the BIR term weight can be simplified. [Robertson and Sparck Jones, 1976] gave four variations of BIR term weight by combining the relevance and occurrence information:

$$F1 \quad : \quad \frac{n_D(t,r)/N_D(r)}{n_D(t,c)/N_D(c)} \tag{2.24}$$

$$F2 \quad : \quad \frac{n_D(t,r)/N_D(r)}{(n_D(t,c)-n_D(t,r))/(N_D(c)-N_D(r))} \tag{2.25}$$

$$F3 \quad : \quad \frac{n_D(t,r)/(N_D(r)-n_D(t,r))}{n_D(t,c)/(N_D(c)-n_D(t,c))} \tag{2.26}$$

$$F4 \quad : \quad \frac{n_D(t,r)/(N_D(r)-n_D(t,r))}{(n_D(t,c)-n_D(t,r))/(N_D(c)-N_D(r)-(n_D(t,c)-n_D(t,r)))} \tag{2.27}$$

F1: only takes account of occurrence probability, and assume the whole collection is non-relevant.

F2: only takes account of occurrence probability.

F3: takes account of both occurrence and non-occurrence probability, and assume the whole collection is non-relevant.

F4: takes account of both occurrence and non-occurrence probability.

Empirical results show that F4 has a better performance. F4 is also called RSJ term weight in some papers. Salton and Robertson's empirical investigation on term relevance weight can be found in [Yu et al., 1982] [Wu and Salton, 1981] [Robertson, 1981] [Robertson and Walker, 1997].

2.4.4 BIR without Relevance Information

The BIR model needs initial relevance information for probability estimation, whereas for real retrieval systems, there is no such relevance information for each query. To carry out the estimation work without relevance information, [Croft and Harper, 1979] assumed that all the query terms have the same probability to occur in the relevant documents, and estimated $P(t_i|\overline{r}) = \frac{n_D(t_i,c)}{N_D(c)}$ (which assumed the whole collection size is far bigger than the relevant collection size, this is also called non-relevant assumption). Then under no relevance information situation, the alternative BIR term weight will be formula 2.30.

$$w(t) = \log \frac{P(t|r)P(\overline{t}|\overline{r})}{P(\overline{t}|r)P(t|\overline{r})} \tag{2.28}$$

$$\sim \log \frac{P(t|r)}{P(\overline{t}|r)} + \log \frac{P(\overline{t}|c)}{P(t|c)} \tag{2.29}$$

$$\sim C + \log \frac{N_D(c) - n_D(t,c)}{n_D(t,c)} \tag{2.30}$$

If $P(t|r)$ is 0.5, which means that a term has the same probability to occur or to be absent from a relevant collection, then constant C will be zero. With very large collection size N ($N \gg n$), the term weight can be approximated as $log\frac{N_D(c)}{n_D(t,c)}$. It is the same as Sparck Jones' inverse document frequency (IDF) [Jones, 1988].

2.4.5 BIR with Little Relevance Information

In relevance feedback retrieval, the system can get relevance judgement for a small portion of documents, or say little relevance information. However, due to the relatively extremely small size of the judged document set, term distributions in the judged documents would not be able to represent the term distributions in the whole collection. Thus, [Robertson and Walker, 1997] suggested: if there is no relevance information, the term weight should based on the inverse collection frequency prior; the term weight should be tunable, and not be negative; if there is a large amount of relevance information, then the term weight should be based on the relevance evidence, and not to take account prior; if there is little amount of relevance information, then the term weight should be the combination of prior and relevance weight; the relevance and non-relevance judgement should be separate.

In their work, the RSJ weight is decomposed into two parts, relevance and non-relevance weight.

$$w(t) = \log \frac{P(t|r)}{1-P(t|r)} - \log \frac{P(t|\overline{r})}{1-P(t|\overline{r})} = w_p(t) - w_q(t) \tag{2.31}$$

Where the prior weights are:

$$w_{p-prior}(t) = k_4 + \log \frac{N_D(c)}{N_D(c) - n_D(t,c)}, \quad w_{q-prior}(t) = \log \frac{n_D(t,c)}{N_D(c) - n_D(t,c)} \tag{2.32}$$

Relevance and non-relevance based weights are:

$$w_{p-rel}(t) = \log \frac{n_D(t,r) + 0.5}{N_D(r) - n_D(t,r) + 0.5}, \quad w_{q-rel}(t) = \log \frac{n_D(t,\overline{r}) + 0.5}{N_D(\overline{r}) - n_D(t,\overline{r}) + 0.5} \tag{2.33}$$

According to the principle of ranking with little relevance, prior weight and relevance weight can be combined, then:

$$w_p(t) = \frac{k_5}{k_5 + N_D(r)} \cdot w_{p-prior}(t) + \frac{N_D(r)}{k_5 + N_D(r)} \cdot w_{p-rel}(t) \tag{2.34}$$

$$w_q(t) = \frac{k_6}{k_6 + N_D(\overline{r})} \cdot w_{q-prior}(t) + \frac{N_D(\overline{r})}{k_6 + N_D(\overline{r})} \cdot w_{q-rel}(t) \tag{2.35}$$

In their experiment, the result showed that the force of relevance weight increases quickly as the relevance set size increases, whilst non-relevant information is not that powerful, but still useful when combined with relevance information.

2.5 Poisson Model

The Poisson model (PM) estimates the term weight with the within document term frequency and the average term frequency in the collection. Although it is not a widely studied model, we would like to introduce it here as we will use it as a bridge between BIR model and language modelling in chapter 3.

$$P(t|c) = P_{Poisson}(tf(t,d) = n|c) = \frac{\lambda^n e^{-\lambda}}{n!} \tag{2.36}$$

Here λ is the average term frequency for term *t*. The probability will be maximal when *n* equals the average term frequency.

With term independence assumption, the Poisson based probability of a document *d* being relevant will be:

$$P(d|q,r) = \prod_{t \in d} P(t|q,r) = \prod_{t \in d} P_{Poisson}(tf(t,d)|q,r) \tag{2.37}$$

The Poisson model might be a little inappropriate in term weighting, as for the same term t, if this term occurs in document n_1 times in d_1, n_2 times in d_2, and $n_1 > n_2 > \lambda(t,r)$, then we intuitively think t is a good discriminator for these two documents, and should weight it higher for document d_1. Unfortunately the Poisson model weights it lower in ($P_{Poisson}(tf(t,d_1) = n_1|r) < P_{PM}(tf(t,d_2) = n_2|r)$), as n_1 deviates further to average TF than n_2. It means t contributes less to d_1 than d_2 in Poisson model. Actually this is not what we expected, as the content words should be favored if they occur more frequently in a document.

To address this problem, odds of $P(d|r)$ may be a good solution. Because if the term t's frequency deviates from average term frequency in the relevant documents, then it tends to deviate more from the average TF in the non-relevant documents.

Follow the BIR model's term independence assumption, and only query term affect the ranking, the Poisson model is:

$$RSV_{PM}(d,q) \quad = \quad O(r|d) = \frac{P(d|r)}{P(d|\overline{r})} = \prod_{t \in d \cap q} \frac{P_{Poisson}(t|r)}{P_{Poisson}(t|\overline{r})} \cdot \prod_{t \in q \setminus d} \frac{P_{Poisson}(t|r)}{P_{Poisson}(t|\overline{r})} \tag{2.38}$$

Then we replace the Poisson probabilities in equation 2.38 with their estimations. When the query term appears in the document, the $P_{Poisson}(tf(t,d) = n|x) = \frac{\lambda(t,x)^{n_L(t,d)} \cdot e^{-\lambda(t,x)}}{n_L(t,d)!}$ When the query term does not appear in the document, the $P_{Poisson}(tf(t,d) = 0|r) = e^{-\lambda(t,x)}$. Here, $\lambda(t,x)$ is term t's average term frequency in document set x, and x can be r or $\overline{r}$ representing relevant or non-relevant document set.

$$RSV_{PM}(d,q) \quad = \quad \prod_{t \in d \cap q} \left(\frac{\lambda(t,r)}{\lambda(t,\overline{r})} \right)^{n_L(t,d)} \cdot \prod_{t \in d \cap q} \frac{e^{-\lambda(t,r)}}{e^{-\lambda(t,\overline{r})}} \cdot \prod_{t \in q \setminus d} \frac{e^{-\lambda(t,r)}}{e^{-\lambda(t,\overline{r})}} \tag{2.39}$$

$$\propto \quad \sum_{t \in d \cap q} \log \left(\frac{\lambda(t,r)}{\lambda(t,\overline{r})} \right)^{n_L(t,d)} + C \tag{2.40}$$

In equation 2.40, C equals $\sum_{t \in q} \log \frac{e^{-\lambda(t,r)}}{e^{-\lambda(t,\overline{r})}}$. C depends on the query but not the document, which can be dropped from the RSV.

In the Poisson model, the distributions of term within the collection and document are both taken into account in the retrieval function.

[Harter, 1975a] found good terms (non-functional) actually tend to have different distribu-

tion in relevant and non-relevant document, and if there is another parameter k describing the probability that a document comes from relevant documents, then the Poisson model can also be formulated as 2-Poisson model shown in equation 2.41, where λ_1 and λ_2 are the average term frequencies in relevant and non-relevant collections.

$$P_{2Poisson}(tf = n) = k\frac{e^{-\lambda_1}\lambda_1^n}{n!} + (1-k)\frac{e^{-\lambda_2}\lambda_2^n}{n!} \tag{2.41}$$

[Margulis, 1992] has looked into N-Poisson model, which divided the whole collection into n sub-collections and assumed each term has a separate Poisson distribution in each sub-collection, with average term frequency λ_i:

$$P_{nPoisson}(tf = n) = \sum_i k_i \frac{e^{-\lambda_i}\lambda_i^n}{n!} \tag{2.42}$$

$$\sum_i k_i = 1$$

He found that most terms or words are 2-Poisson distribution.

2.6 BM25

BM25, best matching function, was developed from BIR model [Robertson et al., 1994] [Jones et al., 2000a][Jones et al., 2000b]. It differs from the BIR model in that it incorporates within document term frequency, query term frequency, document length and various parameters to adjust the term weight. As a whole, BM25 also adjusts RSV with document length and query length.

$$RSV_{BM25}(d,q) = \sum_{t\in q} w(t) + k_2 \cdot ql(q) \cdot \frac{avgdl - dl(d)}{avgdl + dl(d)} \tag{2.43}$$

$$w(t) = s_1 \cdot s_3 \cdot \frac{tf(t,d)^c}{K^c + tf(t,d)^c} \cdot w^{(1)} \cdot \frac{qtf(t,q)}{k_3 + qtf(t,q)}$$

$$K = k_1 \cdot \left((1-b) + b\frac{dl(d)}{avgdl}\right)$$

Notation:

w_t: term weight for the term t.

tf: times that term t occurs in the document d.

qtf: times that term t occurs in the query formula q.

ql: length of the query q.

dl: length of the document d.

$avgdl$: average length of the documents in the collection.

K: TF normalization factor. It varies with respect to document length. K is bigger for long documents than for short ones.

$w^{(1)}$: RSJ term weight function.

s_i: parameters for tuning term weight.

b, c: constants used to adjust TF normalization in term weight. if b and c are set to 1, BM25 will be the BM11; if $c = 1$ and $b = 0$, BM25 will be the BM15. Different values of b will yield different results.

This model includes TF, RSJ term weight, document length, query length and other parameters into the retrieval function. Document length has an impact on TF normalization, as well as the RSV. The formula $tf/(K+tf)$ is the simple approximation of 2-Poisson distribution [Robertson and Walker, 1994]. The second part of BM25 is the document length corrector used to modify the effect of document length on RSV. It is a document level corrector and used only once after all the term weights have been calculated. When the document length is average document length, then the RSV will not be affected. For the document that is longer than average length, its RSV will be decreased slightly. The longer the document, the more will be deducted from the RSV. For the document shorter than average, its RSV will be increased slightly, the shorter the document, the more will be added to the RSV. The reason and techniques of normalization in the retrieval function will be discussed in section 2.10.

In practical application, BM25 model is simplified by setting some parameters to certain values due to the heavy work of parameter estimation. For example, when $c = 1$, $s_1 = 1 + k_1$, $s_3 = 1 + k_3$, BM25 is simplified to formula 2.44:

$$RSV(d,q) = \sum_{t \in q} \frac{(k_1+1)tf(t,d)}{K + tf(t,d)} \cdot w^{(1)} \cdot \frac{(k_3+1)qtf(t,q)}{k_3 + qtf(t,q)} + k_2 \cdot ql(q) \frac{avgdl - dl(d)}{avgdl + dl(d)} \qquad (2.44)$$

2.7 Divergence From Randomness (DFR) Model

The Divergence from Randomness (DFR) is a model measuring the divergence of actual term distribution from random distribution [Amati and van Rijsbergen, 2002]. There are three components in the DFR: information function computing the information that a term contains from the random distribution, risk function adjusting the information gain and term normalization.

$$weight(t,d) = gain(t,d) = P_{risk} \cdot P_{random}(t,d|c) \tag{2.45}$$

- Information Function based on Randomness Models

  The information that a term contain is defined as: "The more the divergence of the within document term frequency of term t from its frequency distribution within the collection, the more the information carried by the term t in the document d"[Amati and van Rijsbergen, 2002].

$$\text{weight}(t,d) \propto -\log P_{random}(t,d|c) \tag{2.46}$$

  In order to estimate the probability that a term occurs in a document *tf* times, different assumption can be applied.

  Binomial distribution:

$$P_{random}(t,d|c) = \begin{pmatrix} tf_c \\ tf_d \end{pmatrix} p^{tf_d}(1-p)^{tf_c - tf_d}, \quad p = 1/N \tag{2.47}$$

  Notation:

  tf_c: the term frequency of the term t in the collection c, also in notation $N_L(t,c)$.

  tf_d: the term frequency of the term t in the document d, also in notation $N_L(t,d)$.

  N: the number of documents in the collection.

  p: the probability term t occurs in the document, $p = 1/N$.

  Geometric distribution:

$$P_{random}(t,d|c) = \left(\frac{1}{1+\lambda}\right) \cdot \left(\frac{\lambda}{1+\lambda}\right)^{tf_d} \tag{2.48}$$

where λ is average term frequency in the collection, $lambda = n_L(t,c)/N_D(c)$.

Term is independent of all other tokens:

$$P_{random}(t,d|c) = \left(\frac{n+0.5}{N+1}\right)^{tf_d} \tag{2.49}$$

where n is the number of document containing the query term, and N is total number of documents in the collection. And $weight(t,d) \propto \log\left(\frac{n+0.5}{N+1}\right)^{tf_d} = tf_d \cdot idf$

- First Normalization for Information Gain

  The aim of first normalization is to smooth the weight assigned according to random process 2.46. Risk function is used to decide the portion of randomness weight that should be assigned to the term. It is also called information gain.

$$gain(t,d) = P_{\text{risk}} \cdot P_{random}(t,d|c) \tag{2.50}$$

  The more the term occurs in the elite set, the less the term frequency is due to randomness, and thus the smaller the probability Risk is, that is:

$$P_{\text{risk-L}} = \frac{1}{tf_d+1} \quad \text{Laplace model} \tag{2.51}$$

$$P_{\text{risk-B2}} \propto \frac{tf_c}{df(tf_c+1)} \quad \text{Ratio } B \text{ of two binomial distributions} \tag{2.52}$$

  where df is the number of documents containing the term.

- Term Frequency Normalization

  Term frequency normalization is also call second normalization in DFR. To sort the problem that long documents tend to have high within document frequency (TF), TF normalization in DFR is:

$$tf_{norm} = tf \cdot \log\left(1 + c \cdot \frac{avgdl}{dl}\right) \tag{2.53}$$

where $avgdl$ is the average document length, and dl is the document length.

DFR is a framework which can accommodate many retrieval models by replacing the variants of the three components. BM25 can also be expressed in this framework, when token independence assumption, Laplace risk model and TF normalization applied.

2.8 Language Modelling

Language Modelling (LM) is another successful model coming from the statistic language modelling area, it ranks a document based on the probability that a query generate the document $P(d|q)$.

By using Bayes theorem, the estimation of the probability that query generates the document can be replaced by the probability that the query occurs in the document (see equation 2.54). The assumption behind this equation is that the probability of a document being relevant to a query is correlated to the probability of the query being generated by the document [Ponte and Croft, 1998a].

$$P(d|q) = \frac{P(q|d) \cdot P(d)}{P(q)} \tag{2.54}$$

Since $P(q)$ is the same for any query, and $P(d)$ is assumed to the same for every document in the whole collection, then:

$$P(d|q) \propto P(q|d) = \prod_{t \in q} P(t|d) \propto \sum_{t \in q} \log P(t|d) \tag{2.55}$$

Certainly there is no necessity to assume that $P(d)$ is constant for the entire document collection, estimating $P(d)$ with document length will lead to different results. [Blanco and Barreiro, 2008, Kamps et al., 2005] all showed that the length prior will improve the retrieval performance.

To avoid the situation that $P(t|d)$ will be set to 0 due to some query terms being absent from the document, a smoothing method is needed. Croft[Ponte and Croft, 1998a], Hiemstra[Hiemstra, 2000] and Zhai [Zhai and Lafferty, 2002] have introduced different smoothing methods. The simplest, yet effective smoothing methods is linear mixture:

$$P(t|d) = (1-\lambda)P(t|c) + \lambda P(t|d) \tag{2.56}$$

Here, $P(t|c)$ is the probability that term t occurs in the collection, it is used to smooth the $P(t|d)$. When term t does not occur in the document, term weight will be decided by the probability that it occurs in the collection. $P(t|c)$ is the same for all the documents because it is estimated by collection frequency.

Language model term weight can also be reformulated as equation 2.57, when it is divided

by a constant $C = \prod_{t \in q}(1-\lambda)P(t|c)$:

$$RSV_{LM}(d,q) = P_{LM}(q|d) = C \cdot \sum_{t \in q \cap d} \log(1 + \frac{\lambda P(t|d)}{(1-\lambda)P(t|c)}) \tag{2.57}$$

Notation:

$P(t|c)$: Probability that a term occurs in the collection. Two kinds of estimations can be used: one is document event space based, $\frac{n_D(t,c)}{N_D(c)}$; the other one is term location space based, $\frac{n_L(t,c)}{N_L(c)}$. $n_L(t,c)$ is the number of times that term t occurrs in collection c, and $N_L(c)$ is the total token number in collection c.

$P(t|d)$: Probability that a term occurs in a document, $P(t|d) = \frac{n_L(t,d)}{N_L(d)}$. $n_L(t,d)$ is the number of times that term t occurrs in document d, and $N_L(d)$ is the total token number in document d.

λ: Parameter used to balance the importance of the $P(t|c)$ or $P(t|d)$. Experience shows that LM works best when λ is approximate 0.2.

2.8.1 Smoothing in LM

In recent years, different smoothing methods have been investigated: Jelinek-Mercer(J-M), Laplace, Dirichlet or Risk based, which can be found in [Ponte and Croft, 1998b, Zhai and Lafferty, 2004, Zaragoza et al., 2003, Hiemstra, 2001]. The followings are widely used smoothing methods:

- Jelinek-Mercer Smoothing

  In Jelinek-Mercer smoothing, the probability that a document generates a query is a linear interpolation of document model and collection model. If the term does not occur in the documents, then this probability only comes from the collection model.

$$\begin{aligned} P_{\text{J-M}}(t|d,c) &= \lambda \cdot P(t|d) + (1-\lambda) \cdot P(t|c) && (2.58) \\ &= \lambda \cdot \frac{n_L(t,d)}{N_L(d)} + (1-\lambda) \cdot \frac{n_L(t,c)}{N_L(c)} && (2.59) \end{aligned}$$

- Laplace Smoothing

  Laplace smoothing is a method which adds one extra count to the term, when α in equation 2.60 equals 1, it is Laplace smoothing. Sometime 1 is not so good in the probability

estimation, so α can be any real positive number, which is termed as Lidstone smoothing.

$$P_{\text{Laplace}}(t|d,c) = \frac{tf(t,d)+\alpha}{|d|+\alpha\cdot V} \quad (2.60)$$

$$= \frac{n_L(t,d)+1}{N_L(d)+V} \quad (2.61)$$

- Dirichlet Smoothing

  Dirichlet Smoothing is a Bayesian smoothing which estimates the maximum posterior probability of a multinomial distribution model[Zaragoza et al., 2003]. The name come from the Dirichlet distribution which a typical multinomial distribution and conjugate prior for Bayesian analysis.

$$P_{\text{Dirichlet}}(t|d,c) = \frac{tf(t,d)+\mu P(t|c)}{|d|+\mu} \quad (2.62)$$

$$= \frac{|d|}{|d|+\mu}\cdot\frac{n_L(t,d)}{N_L(d)} + \frac{\mu}{|d|+\mu}\cdot\frac{n_L(t,c)}{N_L(c)} \quad (2.63)$$

  Dirichlet smoothing can be expressed in linear combination, it is sensitive to collection size and vocabulary size. μ is bigger if the documents contain more rare terms, and smaller if the documents contain more repeated terms. Laplace smoothing can be treated as a special case of Dirichlet smoothing. Here $|d|$ is the document length $N_L(d)$.

There are also some other smoothing methods: risk based smoothing, leave-one-out, absolute discounting, two stage smoothing, etc ([Zhai and Lafferty, 2002] [Ponte and Croft, 1998b] [Generet et al., 1995] [Ney et al., 1994] [Chen and Goodman, 1996] [Chen et al., 2000]), which we do not cover here.

2.8.2 Relevance in LM

When [Ponte and Croft, 1998a] started applying language modelling to information retrieval, there was no explicit relevance variable in LM. Nevertheless, they assumed that users have a reasonable idea of terms that are likely to occur in documents of interest, and will choose query terms that can distinguish documents from others in the collection. They also assumed that probability of relevance and probability of query generation are correlated. From all the assumptions, it can be concluded that the query is generated from the more or less relevant document.

[Lavrenko and Croft, 2001] introduced explicit the relevance variable into LM. They assumed that query is not generated from single document model, but from relevant document set. As the query terms $q_1, q_2, \cdots, q_n$ are the only knowledge about the relevant documents, so the probability that t occurs in the relevant documents can be approximated by equation 2.64

$$P(t|r) = P(t|q_1, q_2, \cdots, q_n) = \frac{P(t, q_1, q_2, \cdots, q_n)}{P(q_1, q_2, \cdots, q_n)} \tag{2.64}$$

There are two ways to estimate $P(t, q_1, q_2, \cdots, q_n)$:

$$P(t, q_1, q_2, \cdots, q_n) = \sum_{M_D \in M} P(M_D) P(t|M_D) \prod_{i=1}^{n} P(q_i|M_D) \tag{2.65}$$

or :

$$P(t, q_1, q_2, \cdots, q_n) = P(t) \prod_{i=1}^{n} P(q_i|t) = P(t) \prod_{i=1}^{n} \sum_{M_D \in M} P(q_i|M_D) P(M_D|t) \tag{2.66}$$

Here, M is a universe of sample of distribution. M_D is a document model, the sample from M, which corresponds to a document.

Different to Lavrenko's relevant document set estimated from query term, [Hiemstra et al., 2004] included limited relevance information from user feedback into the LM mixture.

$$P(t_1, t_2, ...t_i|d) = \prod_i ((1 - \mu - \lambda) P(t_i|c) + \mu P(t_i|r) + \lambda P(t_i|d)) \tag{2.67}$$

And [Azzopardi and Roelleke, 2007] proposed to smooth term probability from both relevant and no-relevant document set, although they did not give out how to estimate the relevant and non-relevant document set.

2.9 Term Weighting and Probabilistic Estimation

2.9.1 Probability Estimation

Probability estimations are also important parts in probabilistic retrieval model, which will lead to different term weights, and rankings of the documents. The followings are the normally used probabilistic models in term weighting:

- Binomial Model

If the event of a term occurring in document is an independent Bernoulli trial with success probability p, then the binomial probability model for the term t occurring in a document k times is:

$$\begin{aligned} P_{Binomial,n,p}(tf(t,d)=k|c) &= \binom{n}{k} p^k q^{n-k} \\ p &= 1/N, \quad q = 1-p = (N-1)/N \end{aligned} \tag{2.68}$$

Here p is the probability that the term occurs in a document, it can be defined as $1/N$; N is total number of documents in the collection; q is the probability that term t won't occur in a document, n is the total term frequency in the document collection $n_L(t,c)$.

- Poisson Model

  The binomial distribution given above can be approximated by Poisson distribution when n is large enough:

$$\begin{aligned} P_{Poisson,\lambda}(tf(t,d)=k|c) &= \lim_{n\to\infty} \binom{n}{k} p^k q^{n-k} = \frac{\lambda^k e^{-\lambda}}{k!} \\ np &= \lambda \end{aligned} \tag{2.69}$$

  In the Poisson distribution, λ means the average term occurrences in a document.

  Document frequency distribution is similar to term frequency. The Poisson model can be used to describe the document frequency. But according to result observed, the predicted IDF value computed based on Poisson model has some distance from observed IDF value especially in the middle value[Church and Gale, 1995].

  Harter proposed 2-Poisson model in [Harter, 1975b], which assumes term has one Poisson distribution in elite set and another Poisson distribution in non-elite set. Each set has its own average term frequency for term t, λ_1 and λ_2 separately.

$$P_{2-Poisson,\lambda_1,\lambda_2}(tf(t,d)=k|c) = \alpha \frac{\lambda_1^n e^{-\lambda_1}}{n!} + (1-\alpha)\frac{\lambda_2^n e^{-\lambda_2}}{n!} \tag{2.70}$$

  Robertson [Robertson et al., 1981] investigated 2-Poisson model, found that it has the same performance as BIR model, and worse than IDF after simplification.

- K-mixture Model

According to the experimental results, Poisson model can not fit the data well, the k-mixture model is better fitting to empirical data. K-mixture is based on the conditional probability of having more than k occurrences in a document, given that exactly k occurrences have already occurred [Katz, 1996].

$$P_{K-mixture}(k) = (1-\alpha)\delta_{k,0} + \frac{\alpha}{\beta+1}(\frac{\beta}{\beta+1})^k \tag{2.71}$$

$$\beta = \lambda 2^{IDF} - 1$$

$$\alpha = \lambda/\beta$$

$\delta_{k,0}$ is 1 when k=0, and 0 otherwise. λ is the average term frequency in the collection $\lambda = N_L(t,c)/N_D(c)$. Continuous K-mixture can be thought as a mixture of Poisson's, when θ corresponds to average occurrence in different set of documents.

$$P_{K-mixture-continue}(k) = \int_0^\infty \phi(\theta)\pi(\theta,k)d\theta \quad for\ \ k = 0,1,2\cdots \tag{2.72}$$

$\phi(\theta)$ is some density function, and $\int_0^\infty \phi(\theta)d\theta = 1$. For each θ, $\pi(\theta,k)$ is the corresponding Poisson distribution function.

2.9.2 Entropy

The concept of entropy originated in physics (the second law of thermodynamics), Shannon introduced it into information theory, where information entropy is a measure of the uncertainty associated with a random variable. It quantifies the information contained in a message, usually in bits. It is the minimum message length necessary to communicate information. In IR, each term can be treated as a signal, then the information contained in the document can be measured by the distribution of the terms.

In this section we brief some concepts of entropy, which will be used in the later sections. More information about entropy can be found in [Thomas M. Cover, 2006]

Entropy

Entropy is a measure of the uncertainty associated with a random variable:

$$H(X) = E(I(X)) = -\sum_{i=1}^{n} P(x_i)\log P(x_i) \tag{2.73}$$

Here I(X) is a measure of the information content associated with the outcome of a random variable, which is $-\log P(x_i)$.

As $1 \geq P(x_i) \geq 0$, $H(X)$ is non-negative. If all $P(x_i)$ are equal, then $H(X)$ is maximal, and the system conveys maximum information. If $P(x_i)$ equals 1 or 0, then $H(X)$ equals 0. Because $0 \cdot \log 0 = 1 \cdot \log 1 = 0$, there is no uncertainty in the system.

The joint entropy H(X,Y) is defined as :

$$H(X,Y) = -\sum_{x \in X} \sum_{y \in Y} P(x,y) \log P(x,y) \tag{2.74}$$

Conditional Entropy

Conditional entropy is the expected value of entropy of the conditional distribution, averaged over the conditional variable.

$$\begin{aligned} H(Y|X) &= \sum_{x \in X} p(x) H(Y|X=x) && (2.75)\\ &= -\sum_{x \in X} \log P(x) \sum_{y \in Y} P(y|x) \log P(y|x) && (2.76)\\ &= -\sum_{x \in X} \sum_{y \in Y} P(x,y) \log P(y|x) && (2.77)\\ & && (2.78) \end{aligned}$$

The relations between conditional entropy and joint entropy is : $H(X,Y) = H(X) + H(Y|X)$.

Relative Entropy

Relative entropy, also known as the Kullback-Leibler divergence, is the distance between two probability mass functions P(x) and Q(x). And it is defined as:

$$D_{KL}(P(X)||Q(X)) = \sum_{x \in X} P(x) \log \frac{P(x)}{Q(x)} \tag{2.79}$$

$D(P||Q) = 0$, if P=Q. Relative entropy is a commonly used measurement for the distance of term being informative in a given model and its real distribution.

Mutual Information

Mutual information is a measure of the amount of information that one variable contains about another one. It is the relative entropy between the joint probability and the product of two marginal probabilities.

$$I(X;Y) = \sum_{x\in X}\sum_{y\in Y} P(x,y)\log\frac{P(x,y)}{P(x)P(y)} \quad (2.80)$$

$$= H(X,Y) - H(X|Y) \quad (2.81)$$

2.9.3 Inverse Document Frequency

After talking about probabilistic models and probability estimation, we will look at the simple and effective term weight scheme: inverse document frequency (IDF). IDF was proposed as term weighting scheme in [Jones, 1988], it measures the importance of a term to a query by counting the number of documents in which the term occurs. It is based on the intuition that good terms should not occur in too many documents, as if a term occurs in too many documents, it is not able to distinguish the documents. Lower term weights should be given to those terms occurring in more documents, and higher weights to the terms occurring in fewer documents.

The definition of IDF is:

$$idf(t,c) = -\log\frac{n_D(t,c)}{N_D(c)} = \log\frac{N_D(c)}{n_D(t,c)} \quad (2.82)$$

Here, $N_D(c)$ is the document number for the collection, $n_D(t,c)$ is the number of documents in which term t occurs with respect to the collection.

Features of IDF ([Robertson, 2004]):

- $\frac{n_D(t,c)}{N_D(c)}$ is the probability that a random document contain term t.
- Operation log makes term weight additive (with the assumption of independent term occurrences).
- IDF can act as a probabilistic model when there is no relevance information.

In early bibliographic retrieval, pure IDF is an effective term weighting scheme. With the appearance of full document retrieval, the term frequency within document (number of a term occurring in a document) was also found to be important to term weight. Currently TF and IDF are involved in almost all term weight schemes, although various normalizations or parameters are applied to emphasize different factors.

Many researchers investigated IDF in both theoretical and empirical aspects after its introduction. [Croft and Harper, 1979] theoretically proved that IDF can be an alternative to

relevance term weight when there is no relevance information and relevant document set size is fairly small compared to the whole document collection. [Greiff, 1998] statistically showed that $\log Odds(t|r)$ is roughly linear to $\log Odds(t|c)$, which supports the conclusion in [Croft and Harper, 1979]. [Wu and Salton, 1981] showed that for medium frequency terms, the differences between their relevance term weight and IDF value are small, and most query terms fall into medium frequency range. They also found that there is not much improvement in retrieval performance by replacing of IDF by relevance term weight.

[Church and Gale, 1995] pointed out that the IDF measurement is more robust than purely Poisson based measurement, but K-mixture method is better than IDF. This is because key words are far from Poisson distribution. The further the distributions are from the Poisson, the move effective the words distinguish the documents. However, IDF is better than ILF although they are similar in some respects, as IDFexpresses the clinginess of the term.

ILF is inverse location frequency, also named inverse collection token frequency (ICTF). The definition of ILF is:

$$ilf(t,c) = -\log \frac{n_L(t,c)}{N_L(c)} \tag{2.83}$$

ILF is different to IDF in that IDF is defined on document space, whilst ILF is defined on location space. It is inversely proportional to the total number of times that the term appears in the document collection. IDF and ILF can sometimes work totally differently, as for the two terms, they can have the same location frequencies, but one may occur in a large number of documents, and the other one occurs in a small amount of documents; or vice versa. Because IDF can show the clinginess of terms, IDF will be better in the aspect discriminating document [Church and Gale, 1995]. However, IDF and ILF can be used in a dual way, which will be introduced in chapter 3.

2.10 Normalization

Normalization is another important aspect of retrieval models. In full document retrieval, the document length has a great impact on the retrieval result, although the relevance of a document to a query is independent of document length. The reason is that a term will occur more times in long documents than in short documents, which leads to a higher TF value; and long documents have a higher probability of matching the query terms because of the larger number of terms in

the long documents. These two aspects enable the long documents to have a higher chance to be retrieved, and with a higher rank. A good normalization will give an equal retrieval chance to each document independent of length. Commonly used normalization methods include: cosine normalization, maximum TF normalization, byte length normalization and pivoted document length normalization.

1. *Cosine Normalization:*

$$w_{cosine\_norm} = \frac{w}{\sqrt{w_1^2 + w_2^2 + \ldots + w_n^2}} \tag{2.84}$$

Cosine normalization decreases the document length effect on term frequency, but it has a bias toward small documents. Here, w_i is the term weight for term t_i.

2. *Max TF Normalization:*

$$tf_{max\_norm} = (1-s) + s \times \frac{tf}{max_{tf}}, 0 \leq s \leq 1 \tag{2.85}$$

This normalization restricts the value of TF from 0 to 1. Different retrieval systems use different s value(SMART-0.5, INQUERY-0.6) to adjust term weight. Sometimes logarithms are used for a large collection. This normalisation solves the problem that a term occurs more times in the long documents, but still has a bias to the long documents, because it can not solve the problem of long documents having more chance to match the query term.

3. *Byte Length Normalization:*

$$RSV(d,q) = \sum_{t \in q} \frac{(k_1+1)tf}{K+tf} \cdot w^{(1)} \cdot \frac{(k_3+1)qtf}{k_3+qtf} + k_2 \cdot ql \frac{avdl - dl}{avdl + dl} \tag{2.86}$$

The above formula is the normalization used in the BM25 model. Document byte length will be used to normalize both TF and the RSV. TF normalization factor K is affected by document length; on top of that, RSV of the document will be normalized by the document length again in order solve the two effects of long documents. (please refer to section 2.6).

4. *Mapping Normalization:*

$$w_{map\_norm} = \lfloor k \cdot \frac{\log w - \log w_{(d,max)}}{\log w_{(d,max)} - \log w_{(d,min)} + \varepsilon} \rfloor + 1 \tag{2.87}$$

The mapping method is a local normalization working on the documents, which maps the weights to the area of 1 to k [Anh and Moffat, 2002]. This method does not concentrate on the value of term weight, it relies more on its rank. The term weights can be mapped to a geometric sequence or an arithmetic sequence.

5. *Pivoted Document Length Normalization:*

$$w_{pivoted\_norm} = \frac{w}{(1-slope) \cdot pivot + slope \cdot old\_norm\_normalization} \tag{2.88}$$

Pivoted document length normalization is thought to have better performance according to the observation of retrieval quality ([Singhal et al., 1996]). It is based on the idea that the probability of retrieving a relevant document should not be influenced by its length, the equivalent chance will help to achieve better retrieval performance. Some other normalizations like cosine normalization, may have bias toward short documents. To make the retrieval relevance close to the real document relevance, sometimes the normalization factor should be increased and sometimes vice verse. Pivot value *pivot* here can be any collection specific value, i.e.. average of old normalization, average document length... After training, the best normalization parameters will be achieved. Pivoted normalization provides a method that can be used to normalize other normalizations in order to decrease the distance between probability of relevance and probability of retrieval. It is similar to the normalization in the BM25 model.

6. *DFR Normalization:* DFR has two steps of normalization: one is for information gain based on random distribution; the other is for observed term frequency according to document length[Amati and van Rijsbergen, 2002]. The latter one is as the following:

$$tf_{norm} = tf \cdot \log_2(1 + c \cdot \frac{avgdl}{dl}) \tag{2.89}$$

where *avgdl* and *dl* are the average length of a document in the collection and the length of the observed document, respectively. Although different in form, the second normalization

of DFR does the same job as BM25 or pivoted document length normalization, penalizing the long document.

All these normalization methods can be combined together or some can be selected to achieve better retrieval results. Smoothing in language modelling plays the same role as normalization, apart from avoiding the zero probability problem.

2.11 Summary

In this section, we have presented some successful retrieval models, term weighting schemes and normalization methods. Most of the models are based on the occurrence probabilities, although each model has its own assumptions. In some models, the different probability estimations are termed identically. It is of great interest to find out whether the probabilistic models are connected and how they are connected? For the heuristic model TF-IDF, how can it be connected to probabilistic models? In the next chapter, we will look into these questions.

Chapter 3

Relationships between Models

The search for the best retrieval model drives information retrieval research. Over the decades, many different types of retrieval models have been proposed and developed. Although some probabilistic models share the same origin, namely the probability of relevance, they differ with respect to event spaces. In this chapter, we investigate in a stringent mathematical formalism the parallel derivation of three grand probabilistic models: binary independent retrieval (BIR) model, Poisson model (PM), and language modelling (LM) from their event spaces, probability estimations, relevance assumptions and ranking rationales, and then draw the theoretical justification from the probabilistic framework.

This chapter is structured as follows. In section 3.1 we list the event spaces, probability estimations, and relevance assumptions of the three grand models. In section 3.2 we show the ranking rationales of the models. In section 3.3 we start from the average characteristic of PM, then show how it bridges BIR model and LM. In section 3.4 we show the dual representation of TF-IDF with BIR or LM parameter. In section 3.5 we decompose $P(d|q,c)$ and $P(q|d,c)$ based on either independent or disjoint terms, then derive TF-IDF formula with appropriate assumptions. In section 3.6 we define a document and query independence (DQI) model according to the disjoint decomposition, and draw a connection to TF-IDF with integral. The summary is briefed in section 3.7.

3.1 Probability Spaces for the IR Models

Probability is the chance or likelihood of something happening. A probability space is expressed by a triple (Ω, F, P). Ω is sample space, which is a set of all possible outcomes for an activity or experiment. F is called events, which are sets of outcomes for which one can ask a probability. Probability measure P is a function which maps event space to real space [0,1], and indicates the chance that event F will happen. In this section the probability spaces of BIR model, PM and LM are clarified.

3.1.1 Sample Space of the Models

BIR Model

Sample space: R.

R: set of relevance judgements $r, \bar{r}$. The relevance judgement is query-based.

As it is hard to assign relevance probability directly, so it is converted to the product of the probability odds of document terms appearing in relevant document set. Hence, the sample space shifts to an alternative one.

Alternative sample space T:

T: set of terms $t_1, t_2, ...t_k$, which independently constitute queries or documents.

PM

Sample space: D.

D: set of documents $d_1, d_2, \cdots, d_n$. The probability of a document d being relevant is the product of the probabilities of document terms appearing in a document f times. Hence, the sample space shifts to an alternative sample space T again.

However, PM takes account of the number of times that the term t appears in a document.

LM

Sample space: Q.

Q: set of queries $q_1, q_2, \cdots, q_m$. The probability that a document d generates a query q is the product of the probabilities of query terms appearing in the document. Again, the sample space will be shift to term space T.

Although the sample spaces for the three models are the same in practice, the statistical processes are different. BIR model checks whether term t appears in a document, and assign t

with relative frequency. PM and LM checks the times of term t appearing in a document, and assign t a probability according to the corresponding distribution model and term frequency.

3.1.2 Probability Estimation

Binary Independence Retrieval Model

BIR Model ranks the document based on the probability of a document relevant to a query $P(r|d,q)$. Due to the difficulty of estimating the relevance probability, BIR model estimates the probability of a document occurring in the relevant document sets $P(d|q,r)$ and non-relevant document sets $P(d|q,\overline{r})$ instead. With Bayes theorem, the odds of a document being relevant is:

$$O(r|d,q) = \frac{P(d|q,r)}{P(d|q,\overline{r})} \tag{3.1}$$

With a binary representation of a document, and the assumption that non-query terms have the same probability of occurring in relevant and non-relevant document, $O(r|d,q)$ will be:

$$O(r|d,q) = \frac{P(d|q,r)}{P(d|q,\overline{r})} = \left[\prod_{t\in d\cap q}\frac{P(t|q,r)}{P(t|q,\overline{r})}\right]\cdot\left[\prod_{t\in q-d}\frac{P(\overline{t}|q,r)}{P(\overline{t}|q,\overline{r})}\right] \tag{3.2}$$

$$= \left[\prod_{t\in d\cap q}\frac{P(t|q,r)\cdot P(\overline{t}|q,\overline{r})}{P(t|q,\overline{r})\cdot P(\overline{t}|q,r)}\right]\cdot\left[\prod_{t\in q}\frac{P(\overline{t}|q,r)}{P(\overline{t}|q,\overline{r})}\right] \tag{3.3}$$

$$\overset{rank}{=} \prod_{t\in d\cap q}\frac{P(t|q,r)\cdot P(\overline{t}|q,\overline{r})}{P(t|q,\overline{r})\cdot P(\overline{t}|q,r)} \tag{3.4}$$

As the second part in equation 3.3 is a query dependent function, will be the same for any document. Thus it won't affect the ranking and can be dropped out.

Actually not all non-query terms will not have the same occurrence probabilities in the relevant and non-relevant document set, as some non-query terms may be about the query. As a result, the occurrence of these non-query terms will affect the ranking. Ranking function incorporating the non-query terms is:

$$O(r|d,q) = \frac{P(d|q,r)}{P(d|q,\overline{r})} = \left[\prod_{t\in d\cap q}\frac{P(t|q,r)}{P(t|q,\overline{r})}\right]\cdot\left[\prod_{t\in d-q}\frac{P(\overline{t}|q,r)}{P(\overline{t}|q,\overline{r})}\right] \tag{3.5}$$

$$= \left[\prod_{t\in d\cap q}\frac{P(t|q,r)\cdot P(\overline{t}|q,\overline{r})}{P(t|q,\overline{r})\cdot P(\overline{t}|q,r)}\right]\cdot\left[\prod_{d}\frac{P(\overline{t}|q,r)}{P(\overline{t}|q,\overline{r})}\right] \tag{3.6}$$

The second part in equation 3.6 is a document dependent function, which will be different for each document.

For the probabilities of a term appearing in a relevant set r or non-relevant set r, it will be estimated as:

$$P(t|r) = \frac{n_D(t,r)}{N_D(r)}, \quad P(\overline{t}|r) = 1 - P(t|r) \tag{3.7}$$

$$P(t|\overline{r}) = \frac{n_D(t,c) - n_D(t,r)}{N_D(c) - N_D(r)} \tag{3.8}$$

Poisson Model

PM estimates the probability of observing term t with f times in a document based on term Poisson distribution, and the expectation of the times to observe t in a document: λ . With term independence assumption, $P(r|d,q) = \prod_{t \in q} P(f_t|r)$, where f_t is term t's within-document frequency.

The Poisson estimation of $P_{PM}(f_t|r)$ is as follows:

$$P_{PM}(f_t|r) = \frac{\lambda(t,r)^{f_t}}{f_t!} \cdot e^{-\lambda(t,r)}, \qquad \lambda(t,r) = \frac{n_L(t,r)}{n_D(t,r)} \tag{3.9}$$

As in the BIR model, PM uses odds to rank the documents:

$$P(d|q,r) \quad = \quad \prod_{t \in d \cap q} P_{PM}(f_t|q,r) \cdot \prod_{t \in q \setminus d} e^{-\lambda(t,r)} \tag{3.10}$$

$$O(d|q,r) \quad \stackrel{rank}{=} \quad \prod_{t \in d \cap q} \left(\frac{\lambda(t,r)}{\lambda(t,\overline{r})} \right)^{n_L(t,d)} \tag{3.11}$$

Language Modelling

LM judges the relevance of document to a query by the probability that a document generates the query. Again, term independence assumption is applied here.

$$P(d|q) = \frac{P(q|d) \cdot P(d)}{P(q)}) \stackrel{rank}{=} P(q|d) = \prod_{t \in q} P_{J-M}(t|d) \tag{3.12}$$

Here the assumption is that $P(d)$ is the same for any document.

For the smoothing of $P(t|d)$, there are many different methods which have been introduced in section 2.8. We follow the linear mixture method, with 0.2 for the classical λ setting in this chapter.

$$P_{J-M}(t|d) = \lambda \cdot P(t|d) + (1-\lambda) \cdot P(t|c) \tag{3.13}$$

$$P(t|d) = \frac{n_L(t,d)}{N_L(d)}, \qquad P_(t|c) = \frac{n_L(t,c)}{N_L(c)} \tag{3.14}$$

3.1.3 Relevance Assumptions

Relevance assumptions impact the probability estimations, consequently the ranking of documents. The assumptions in the following sections will help to interpret the probability estimations in different models.

Non-relevance Assumption

BIR model and PM assume that the whole collection is a non-relevant collection. This is because when there is no relevance information, [Croft and Harper, 1979] assumed that each term has the same probability of occurring in the relevant documents, and also assumed that the non-relevant documents are a very large portion of the collection, which makes it reasonable to estimate the occurrence probability in non-relevant collection by occurrence probability in the whole collection. In other words, the whole collection is a non-relevant collection. This assumption leads BIR term weight to $c + \log \frac{N_D(c)-n_D(t,c)}{n_D(t,c)}$, [Robertson, 2004] further assumed the probability that a term occurs in relevant documents is 0.5, then BIR weight is $\log \frac{N_D(c)-n_D(t,c)}{n_D(t,c)}$. PM uses the same way to estimate the average term frequency in the non-relevant set. This is the so called non-relevance assumption.

Relevance Assumption

Language Modelling implicitly assumes all the documents in the collection are relevant documents, which estimates $P(d|q,r)$ by $P(q|d,c)$. This is because [Ponte and Croft, 1998b] assumed that the users can properly formulate their information need, and each document is a document model to generate the query.

Relevance and Non-relevance Assumption

[Lafferty and Zhai, 2003] included both relevant and non-relevant concept in LM formula mathematically, but dropped out the non-relevant based on the assumption that document and query are independent given the condition of non-relevance. [Azzopardi and Roelleke, 2007] proposed to combine two document model given it is being relevance or non-relevant. Nevertheless, they did not give out the solution how to decide whether the document is being relevant or not.

$$P(q|d) = P(q|d,r) \cdot P(r|d) + P(q|d,\overline{r}) \cdot P(\overline{r}|d) \tag{3.15}$$

3.2 Ranking Rationales

BIR: Terms that occur more often in relevant than in non-relevant documents have a positive effect on the RSV, whereas terms that occur less often in relevant than in non-relevant documents have a negative effect on the RSV. Mathematically, we summarize this as follows: $P_{BIR}(t|r) > P_{BIR}(t|\overline{r})$: good term, positive effect on RSV. $P_{BIR}(t|r) < P_{BIR}(t|\overline{r})$: bad term, negative effect on RSV. $P_{BIR}(t|r) = P_{BIR}(t|\overline{r})$: neutral term, no effect on RSV.

With logarithms of odds, good terms have positive term weights, bad terms have negative term weights and neutral terms have 0 weights. However, the negative weight seems to penalize too much the documents with the bad term, as our work in [Roelleke and Wang, 2007] showed that better performance was achieved when replacing the negative weight with 0.

PM: Terms, whose average occurrence in relevant documents is higher than the average occurrence in non-relevant documents, have a positive effect on the RSVs. Whereas terms, whose average occurrence in relevant documents is less than the average occurrence non-relevant documents, have a negative effect on the RSVs. Mathematically, we summarize this as follows: $\lambda_{PM}(t,r) > \lambda_{PM}(t,\overline{r})$: good term, positive effect on RSV. $\lambda_{PM}(t,r) < \lambda_{PM}(t,\overline{r})$: bad term, negative effect on RSV. $\lambda_{PM}(t,r) = \lambda_{PM}(t,\overline{r})$: neutral term, no effect on RSV. The occurrence within the document $n_L(t,d)$ increases the effect of a term.

The probability $P_{PM}(d|q,r)$ is maximal for a document that represents the average term occurrences in the relevant documents, and the probability $P_{PM}(d|q,\overline{r})$ is maximal for a document that represents the average term occurrences in the non-relevant documents. However, the maximal odds is not at the point of average term occurrences.

LM: Large (small) $P(t|d)$ implies strong (little) effect on RSV. Small (large) $P(t|c)$ implies strong (little) effect on RSV. A document containing rare terms (small $P(t|c)$) will have a higher RSV.

3.3 Poisson Bridge BIR and LM

This section shows that PM can be viewed as a bridge connecting BIR and LM. Term's average occurrence connects the document-based and location-based probabilities. This relationship is obtained from rewriting $\lambda(t,c) = \lambda(t,c)$. since $\lambda(t,c) := n_L(t,c)/N_D(c)$, it can be written as:

$$n_L(t,c)/N_D(c) = n_L(t,c)/N_D(c) \tag{3.16}$$

And multiply $1 = \frac{n_D(t,c)}{n_D(t,c)}$ and $1 = \frac{N_L(c)}{N_L(c)}$ on each side,

$$\frac{n_L(t,c)}{N_D(c)} \cdot \frac{n_D(t,c)}{n_D(t,c)} = \frac{n_L(t,c)}{N_D(c)} \cdot \frac{N_L(c)}{N_L(c)} \tag{3.17}$$

Replace $P_{BIR}(t|c) := n_D(t,c)/N_D(c)$[1], and $P_{LM}(t|c) := n_L(t,c)/N_L(c)$[2] into Equation 3.17, we get:

$$P_{BIR}(t|c) \cdot \frac{n_L(t,c)}{n_D(t,c)} = P_{LM}(t|c) \cdot \frac{N_L(c)}{N_D(c)} \tag{3.18}$$

The two fractions in the equation have the following meaning: $avgdl(c) := N_L(c)/N_D(c)$ is the average document length, and $avgtf(t,c) := n_L(t,c)/n_D(t,c)$ is the average term frequency of term t in the all documents containing t.

With the definitions of $avgdl(c)$ and $avgtf(t,c)$, we obtain the following equation connecting BIR and LM:

$$P_{BIR}(t|c) \cdot avgtf(t,c) = P_{LM}(t|c) \cdot avgdl(c) \tag{3.19}$$

We refer to this equation as Poisson bridge , since the equation $\lambda(t,c) = \lambda(t,c)$ with the Poisson parameter $\lambda(t,c)$ leads to the connection of BIR and LM probabilities.

For example, if the term "*sailing*" occurs in 5 locations and 4 documents ($n_L(sailing,c) = 5$, $n_D(sailing,c) = 4$), and the collection has 100 locations and 10 documents ($N_L(c) = 100$, $N_D(c) = 10$). Then, the average within-document frequency of sailing is $\lambda_{PM}(sailing,c) = 5/10$, $avgtf(sailing,c) = 5/4$ locations containing *sailing* per sailing document, and $avgdl(c) = 100/10$ locations per document are expected. Here, $avgtf(sailing,c) < avgdl(c)$, and this is the usual case for most of the terms.

From equation 3.19, it can be concluded that if and only if the average term frequency $avgtf(t,c)$ is less (greater) than the average document length $avgdl(c)$, then the probability $P_{BIR}(t|c)$ is greater (less) than the probability $P_{LM}(t|c)$.

$$\begin{aligned} avgtf(t,c) < avgdl(c) &\iff P_{BIR}(t|c) > P_{LM}(t|c) \\ avgtf(t,c) > avgdl(c) &\iff P_{BIR}(t|c) < P_{LM}(t|c) \end{aligned}$$

This means that for most of the terms, $avgtf(t,c)$ is less than $avgdl(c)$, their $P_{BIR}(t|c)$ is greater than $P_{LM}(t|c)$. Only in the extreme case that a term occurs in very few extremely long

[1] $P_{BIR}(t|c)$ is also noted as $P_D(t|c)$ somewhere.
[2] $P_{LM}(t|c)$ is also noted as $P_L(t|c)$ somewhere.

documents, and the term is repeated throughout the documents, then term will have $P_{BIR}(t|c)$ smaller than $P_{LM}(t|c)$.

3.4 TF-IDF's Explanation with BIR and LM

In this section, the TF-IDF retrieval function will be derived from RSV of PM, with the probabilities $P_{BIR}(t|c)$ or $P_{LM}(t|c)$ respectively, whereas the two formulae are equivalent. These formulae stress the duality of BIR and LM: Although there are difference in event space, parameter estimation, both $P_{BIR}(t|c)$ and $P_{LM}(t|c)$ can be alternatively used for expressing RSV_{PM} and TF-IDF respectively.

3.4.1 Dual Application of BIR and LM Parameters

In this section we will show how to formulate PM with BIR or LM parameters. Let's start from the following $RSV_{PM}(d,q)$ introduced in section 2.5:

$$RSV_{PM}(d,q) := \sum_{t \in d \cap q} n_L(t,d) \cdot \log \frac{\lambda(t,r)}{\lambda(t,\overline{r})} \tag{3.20}$$

Insert the definitions $\lambda(t,r) := n_L(t,r)/N_D(r)$, and $\lambda(t,\overline{r}) := n_L(t,c)/N_D(c)$, i.e. the definitions for the average term occurrence, also multiply $\lambda(t,r)$ and $\lambda(t,\overline{r})$ with $1 = n_D(t,r)/n_D(t,r)$ and $1 = n_D(t,c)/n_D(t,c)$ respectively,

$$RSV_{PM}(d,q) \quad = \quad \sum_{t \in d \cap q} n_L(t,d) \cdot \log \frac{\frac{n_L(t,r)}{N_D(r)} \cdot \frac{n_D(t,r)}{n_D(t,r)}}{\frac{n_L(t,c)}{N_D(c)} \cdot \frac{n_D(t,c)}{n_D(t,c)}} \tag{3.21}$$

Then, insert the definitions $P_{BIR}(t|x) := n_D(t,x)/N_D(x)$, $avgtf(t,x) := n_L(t,x)/n_D(t,x)$, where x is the set of relevant documents, or the whole collection. This leads to the following BIR-based formulation of RSV_{PM}, which is equivalent to the starting point in equation 3.20.

$$RSV_{PM}(d,q) \quad = \quad \sum_{t \in d \cap q} n_L(t,d) \cdot \log \frac{P_{BIR}(t|r) \cdot avgtf(t,r)}{P_{BIR}(t|c) \cdot avgtf(t,c)} \tag{3.22}$$

Similarly by multiplying $\lambda(t,r)$ by $1 = n_L(t,r)/n_L(t,r)$, and $\lambda(t,\overline{r})$ by $1 = n_L(t,c)/n_L(t,c)$, and replacing $n_L(t,x)/N_L(x)$ with $P_{LM}(t,x)$ and $N_L(x)/N_D(x)$ with $avgdl(x)$, we have another RSV_{PM} expressed by P_{LM} parameter.

$$RSV_{PM}(d,q) = \sum_{t \in d \cap q} n_L(t,d) \cdot \log \frac{P_{LM}(t|r) \cdot avgdl(r)}{P_{LM}(t|c) \cdot avgdl(c)} \tag{3.23}$$

The BIR-based and LM-based formulae of RSV_{PM} (Equation 3.23, 3.23) show that both BIR and LM parameters can be used in a dual way.

3.4.2 Dual Representation with IDF and ILF

When we apply the definition $idf(t,x) := -\log P_{BIR}(t|x)$ to the BIR-based equation 3.22, we obtain equation 3.24

$$RSV_{PM}(d,q) = \sum_{t \in d \cap q} n_L(t,d) \cdot \left(idf(t,c) - idf(t,r) + \log \frac{avgtf(t,r)}{avgtf(t,c)} \right) \tag{3.24}$$

With the inverse location frequency (ILF) analogous to IDF: $ilf(t,x) := -\log P_{LM}(t|x)$, we obtain equation 3.25 from the LM-based equation 3.23:

$$RSV_{PM}(d,q) = \sum_{t \in d \cap q} n_L(t,d) \cdot \left(ilf(t,c) - ilf(t,r) + \log \frac{avgdl(r)}{avgdl(c)} \right) \tag{3.25}$$

These IDF-based and ILF-based formulae show that the PM proposes to correct classical TF-IDF by a factor. For IDF, this corrector is IDF within the relevant collection and average term frequency; For ILF, this corrector is ILF within the relevant collection and average document length. The BM25, pivoted document length and many other experiments confirmed that taking into account normalization factors does improve retrieval quality compared to basic TF-IDF. The summation of $idf(t,c) - idf(t,r)$ coincides with the formula in [de Vries and Roelleke, 2005] and F1 weight in [Robertson and Sparck Jones, 1976].

3.5 Explanation of TF-IDF with Term Disjointness or Independence Assumption

In the previous section, we represent PM as TF-IDF with with BIR and LM parameters, including average normalization. In this section, we will explain TF-IDF starting from either a term disjointness or independence assumption.

3.5.1 Independent Terms: P(q|d,c)

In this section, we show that the components of LM term weight correspond to component TF-IDF term weight, but there is some distance to TF-IDF formula.

"Pure" and mixed estimates

The purely location-based LM term weight is captured in the following definition:

$$LLw_{LM}(t,d,c) := 1 + \frac{\delta}{1-\delta} \cdot \frac{P_L(t|d)}{P_L(t|c)} \tag{3.26}$$

The prefix *LL* indicates the respective event spaces. For both, $P(t|d)$ and $P(t|c)$, location-based probabilities are applied. [Hiemstra, 2000] involves a mix of event spaces, and the prefix *LD* indicates this in the next definition:

$$LDw_{LM}(t,d,c) := 1 + \frac{\delta}{1-\delta} \cdot \frac{P_L(t|d)}{P_D(t|c)} \tag{3.27}$$

The Poisson bridge (equation 3.18) relates the purely location-based estimate in equation 3.26 and the location-document-based mix in equation 3.27.

Inserting $P_L(t|c) = \frac{avgtf(t,c)}{avgdl(c)} \cdot P_D(t|c)$ into equation 3.26 injects $P_D(t|c)$ into the purely location-based estimate:

$$LLw_{LM}(t,d,c) = 1 + \frac{\delta \cdot avgdl(c)}{(1-\delta) \cdot avgtf(t,c)} \cdot \frac{P_L(t|d)}{P_D(t|c)} \tag{3.28}$$

Analogously, inserting $P_D(t|c) = \frac{avgdl(c)}{avgtf(t,c)} \cdot P_L(t|c)$ into equation 3.27 injects $P_L(t|c)$ into the document-location mix:

$$LDw_{LM}(t,d,c) = 1 + \frac{\delta \cdot avgtf(t,c)}{(1-\delta) \cdot avgdl(c)} \cdot \frac{P_L(t|d)}{P_L(t|c)} \tag{3.29}$$

The above solution regarding the event space mix has a significant impact on the understanding and validity of parameter estimation and interpretation. The location-document mix is from a probabilistic semantics point of view difficult to justify, while the location-based formulation can solve this problem. For the location-document mix, the setting of the mixture parameter δ can be viewed as the fix that transfers the location-document mix into the location-based formulation. This solution supports that LM could be viewed as a probabilistic interpretation of TF-IDF, however, this interpretation is still distant to the genuine TF-IDF. Therefore, the next section discusses the LM-like decomposition of $P(d|q)$, which leads to a conclusive interpretation of TF-IDF.

3.5.2 Independent Terms: P(d|q,c)

In this section, we show that TF-IDF is contained in the $P(d|q,c)$ based on term independence assumption.

Starting from represent $P(d|q,c)$ with independent terms:

$$P(d|q,c) = \prod_{t \in d} P(t|q,c)^{n_L(t,d)} \tag{3.30}$$

The first transformation splits the product over document terms into two products: a product over document and query terms, and a product over document-only terms.

$$P(d|q,c) = \left[\prod_{t \in d \cap q} P(t|q,c)^{n_L(t,d)} \right] \cdot \left[\prod_{t \in d-q} P(t|q,c)^{n_L(t,d)} \right] \tag{3.31}$$

Next, a query/non-query term assumption is specified, and this assumption is applied to estimate the unknown term probability $P(t|q,c)$.

For query terms: $P(t|q,c) = P(t|q)$. For non-query terms: $P(t|q,c) = P(t|c)$.

This assumption can be viewed as a radical mixture $P(t|q,c) = \delta P(t|q) + (1-\delta)P(t|c)$, this corresponds for query terms to $\delta = 1$, and for non-query terms to $\delta = 0$. The next equation builds on this assumption; $P(t|q,c)$ is replaced by $P(t|q)$ and $P(t|c)$, respectively.

$$P(d|q,c) = \left[\prod_{t \in d \cap q} P(t|q)^{n_L(t,d)} \right] \cdot \left[\prod_{t \in d-q} P(t|c)^{n_L(t,d)} \right] \tag{3.32}$$

Equation 3.32 is multiplied with $1.0 = \prod_{t \in d \cap q} \left[\frac{P(t|c)}{P(t|c)} \right]^{n_L(t,d)}$. Through this, the product over document-only terms (right product) becomes a product over all document terms.

$$P(d|q,c) = \left[\prod_{t \in d \cap q} \left(\frac{P(t|q)}{P(t|c)} \right)^{n_L(t,d)} \right] \cdot \left[\prod_{t \in d} P(t|c)^{n_L(t,d)} \right] \tag{3.33}$$

The right product corresponds to $P(d|c)$. We move this document normalization factor to the left side of the equation, and it prepares for establishing an analogy to the probabilistic odds and TF-IDF.

$$\frac{P(d|q,c)}{P(d|c)} = \prod_{t \in d \cap q} \left(\frac{P(t|q)}{P(t|c)} \right)^{n_L(t,d)} \tag{3.34}$$

In the next two steps, we insert separately the document-based and location-based estimation of $P(d|c)$, $P(t|q)$, and $P(t|c)$. The document-based estimation could be viewed as "incorrect" since the starting point is independent terms; however, the tradition in IR to mix document-based and location-based estimates is a significant rationale to investigate document-based estimates.

Document-based

Inserting the document-based probabilities into equation 3.34 yields:

$$\frac{P(d|q,c)}{P_D(d|c)} = \prod_{t \in d \cap q} \left(\frac{P_D(t|q)}{P_D(t|c)} \right)^{n_L(t,d)} \tag{3.35}$$

Then, the logarithmic form is as follows:

$$\log \frac{P(d|q,c)}{P_D(d|c)} = \sum_{t \in d \cap q} n_L(t,d) \cdot [-\log P_D(t|c) + \log P_D(t|q)] \tag{3.36}$$

This leads to a document-based TF-IDF interpretation:

$$\log P(d|q,c) - \sum_{t \in d} (n_L(t,d) \cdot idf(t,c)) = \sum_{t \in d \cap q} n_L(t,d) \cdot [idf(t,c) - idf(t,q)] \tag{3.37}$$

For independent terms, $P(d|q,c)$, and document-based probabilities, TF-IDF assumes $idf(t,q) = 0$, i.e. $P_D(t|q) = 1$.

This is an exciting interpretation of TF-IDF. The discriminativeness expressed by $idf(t,c)$ is combined with the query-specific discriminativeness $idf(t,q)$, and for $idf(t,q) = 0$, equation 3.37 uncovers TF-IDF. The interpretation of the term probability $P_D(t|q)$ in the query is to view the query as a structured document; frequent terms occur in every part of the query, $idf(t,q) = 0$, and for non-frequent terms, $idf(t,q) > 0$.

The component $\log P_D(d|c)$ can be viewed as fixed document prior if we assume uniform probability for each document. Or the component $\log P_D(d|c) = \sum_{t \in d} n_L(t,d) \cdot idf(t,c)$ is the query-independent TF-IDF value of the document. $P_D(d|c)$ is high for documents that contain discriminative terms. This prior corresponds to the normalization proposed for the vector-space model.

Location-based

Inserting the location-based probabilities into equation 3.34 yields:

$$\frac{P(d|q,c)}{P_L(d|c)} = \prod_{t \in d \cap q} \left(\frac{P_L(t|q)}{P_L(t|c)} \right)^{n_L(t,d)} \tag{3.38}$$

Equation 3.38 corresponds to equation 3.35. To approach TF-IDF, the Poisson bridge $P_L(t|c) = \frac{avgtf(t,c)}{avgdl(c)} \cdot P_D(t|c)$ injects the document-based probability $P_D(t|c)$. This leads to the next equation:

$$\frac{P(d|q,c)}{P_L(d|c)} = \prod_{t \in d \cap q} \left(\frac{avgdl(c)}{avgtf(t,c)} \cdot \frac{P_L(t|q)}{P_D(t|c)} \right)^{n_L(t,d)} \tag{3.39}$$

The logarithmic transformation yields:

$$\log P(d|q,c) - \log P_L(d|c) = \sum_{t \in d \cap q} n_L(t,d) \cdot \left[-\log P_D(t|c) + \log \left(P_L(t|q) \cdot \frac{avgdl(c)}{avgtf(t,c)} \right) \right] \tag{3.40}$$

Equation 3.40 corresponds to equation 3.36, and the final step applies $idf(t,c) = -\log P_D(t|c)$ to uncover TF-IDF.

$$\log P(d|q,c) - \sum_{t \in d} n_L(t,d) \cdot ilf(t,c) = \sum_{t \in d \cap q} n_L(t,d) \cdot \left[idf(t,c) + \log \left(P_L(t|q) \cdot \frac{avgdl(c)}{avgtf(t,c)} \right) \right] \tag{3.41}$$

For independent terms, $P(d|q,c)$, and location-based probabilities, TF-IDF assumes $avgtf(t,c) = P_L(t|q) \cdot avgdl(c)$, which is $P_L(t|q) = P_L(t|d)$.

The location-based equation 3.41 corresponds to the document-based equation 3.37.

The component $\log P_L(d|c) = \sum_{t \in d} n_L(t,d) \cdot ilf(t,c)$ is the query-independent TF-IDF value of the document. $P_L(d|c)$ is high for documents that contain discriminative terms. This prior corresponds to the normalization proposed for the vector-space model.

3.5.3 Independent Terms: O(r|d,q)

In this section, we show the TF-IDF is also a intrinsic part of probabilistic odds of document and query being relevant $O(r|d,q)$.

For independent terms, the following sequence of equations decomposes the probabilistic odds:

$$O(r|d,q) = \frac{P(r|d,q)}{P(\overline{r}|d,q)} \stackrel{rank}{=} \frac{P(d|q,r)}{P(d|q,\overline{r})} \tag{3.42}$$

$$= \prod_{t \in d} \left(\frac{P(t|q,r)}{P(t|q,\overline{r})} \right)^{n_L(t,d)} \tag{3.43}$$

The ranking equivalence in 3.42 follows from Bayes' theorem $P(r|d,q) = \frac{P(d|q,r) \cdot P(q,r)}{P(d,q)}$. Then, $P(q,r)$ can be dropped since it is document-independent.

Equation 3.43 views d as a sequence of conditionally independent term events; thereby, the frequency (multiple occurrence) of a term is captured by the exponent $n_L(t,d)$.

The next transformation reflects the non-query-term assumption: for non-query terms, $P(t|q,r) = P(t|q,\bar{r})$ is assumed, i.e. non-query terms occur in relevant documents as they occur in non-relevant documents. Through this, the product over all document terms reduces to the product over document and query terms ($t \in d \cap q$). A softer approach is to assume $P(t|q,r)/P(t|q,\bar{r})$ to be a constant for the non-query terms ([Croft and Harper, 1979]). For convenience, we omit q in the conditional from this point. This is consistent, since r implies q.

$$O(r|d,q) = \prod_{t \in d \cap q} \left(\frac{P(t|r)}{P(t|\bar{r})} \right)^{n_L(t,d)} \tag{3.44}$$

The right side of equation 3.44 shows a strong analogy to the right side of equation 3.34 ($P(d|q,c)/P(d|c)$). For $r = q$ and $\bar{r} = c$, they are equivalent! This is reasonable since the relevant documents can be viewed as the query, and viewing the collection as an approximation of non-relevant documents is common.

The next sections concern the document-based and location-based estimates of $P(t|r)$ and $P(t|\bar{r})$.

Document-based

The document-based estimate is:

$$O(r|d,q) = \prod_{t \in d \cap q} \left[\frac{P_D(t|r)}{P_D(t|\bar{r})} \right]^{n_L(t,d)} \tag{3.45}$$

$$\stackrel{rank}{=} \sum_{t \in d \cap q} n_L(t,d) \cdot \log \frac{P_D(t|r)}{P_D(t|\bar{r})} \tag{3.46}$$

From the document-based estimate in equation 3.46 and the definition of IDF, the ranking equivalence follows:

$$O(r|d,q) = \sum_{t \in d \cap q} n_L(t,d) \cdot [idf(t,\bar{r}) - idf(t,r)] \tag{3.47}$$

For independent terms, $O(r|d,q)$, and document-based probabilities, TF-IDF assumes $idf(t,r) = 0$, i.e. $P_D(t|r) = 1$.

For $\bar{r} = c$ and $r = q$, the odds-based interpretation in equation 3.47 is equivalent to the interpre-

tation based on $P(d|q,c)$ in equation 3.37. This result also shows why query term probabilities can be estimated from the set of relevant documents.

Location-based

The location-based estimate is:

$$O(r|d,q) = \prod_{t \in d \cap q} \left[\frac{P_L(t|r)}{P_L(t|\overline{r})} \right]^{n_L(t,d)} \tag{3.48}$$

$$\stackrel{rank}{=} \sum_{t \in d \cap q} n_L(t,d) \cdot \log \frac{P_L(t|r)}{P_L(t|\overline{r})} \tag{3.49}$$

For the location-based probabilities, equation 3.44 and the Poisson bridge (equation 3.18) lead to:

$$O(r|d,q) = \sum_{t \in d \cap q} n_L(t,d) \cdot \left[idf(t,\overline{r}) + \log \left(P_L(t|r) \cdot \frac{avgdl(\overline{r})}{avgtf(t,\overline{r})} \right) \right] \tag{3.50}$$

For independent terms, $O(r|d,q)$, and location-based probabilities, TF-IDF assumes $avgtf(t,\overline{r}) = P_L(t|r) \cdot avgdl(\overline{r})$.

Again, for $r = q$ and $\overline{r} = c$, the odds-based interpretation in equation 3.50 is equivalent to the interpretation based on $P(d|q,c)$ in equation 3.41.

The theorem of the total probability allows us to decompose the probability $P(h)$ for a set of disjoint and exhaustive events $e_1, \ldots, e_n$ (i.e. $P(e_i \wedge e_j) = 0$ and $1.0 = \sum_i P(e_i)$).

$$P(h) = \sum_i P(h|e_i) \cdot P(e_i) \tag{3.51}$$

For the event (hypothesis) h being a document or query, and for events (evidence) e_i being terms, the decomposition of $P(q|d)$ and $P(d|q)$ follows. and this decomposition can be viewed as the *disjunctive* alternative to the *conjunctive* alternative when assuming independent terms. From the next section,term are treated as disjoint events. We will represent the relevance probabilities ($P(q|d,c)$, $P(d|q,c)$ and $P(d,q|c)$) based on disjoint terms. Although disjointness assumption does not lead to exact TF-IDF, TF and $\frac{1}{P(t|c)}$ do appear in the formulae.

3.5.4 Disjoint Terms: P(q|d,c)

For $P(q|d)$, the decomposition via disjoint terms yields:

$$P(q|d) = \sum_{t\in q} P(q|t)\cdot P(t|d) \tag{3.52}$$

$$= \frac{1}{P(d)}\cdot\sum_{t\in q} P(q|t)\cdot P(d|t)\cdot P(t) \tag{3.53}$$

$$= P(q)\cdot\sum_{t\in q} P(t|q)\cdot P(t|d)\cdot\frac{1}{P(t)} \tag{3.54}$$

When the collection c is explicit included in the formula:

$$P(q|d,c) = \sum_{t\in q} P(q|t,c)\cdot P(t|d,c) \tag{3.55}$$

$$=\frac{1}{P(d|c)}\cdot\sum_{t\in q} P(q|t,c)\cdot P(d|t,c)\cdot P(t|c) \tag{3.56}$$

$$=P(q|c)\cdot\sum_{t\in q} P(t|q,c)\cdot P(t|d,c)\cdot\frac{1}{P(t|c)} \tag{3.57}$$

The equations 3.56 and 3.57 reflect the application of Bayes' theorem for $P(t|d,c)$ and $P(q|t,c)$, respectively. Inserting $P(t|d,c)=\frac{P(d|t,c)\cdot P(t|c)}{P(d|c)}$ into equation 3.55 yields equation 3.56. Similarly, inserting $P(q|t,c)=\frac{P(t|q,c)\cdot P(q|c)}{P(t|c)}$ into equation 3.55 yields equation 3.57. Notably, $\frac{1}{P(t|c)}$ in equation 3.57 lacks a probabilistic interpretation, but the derivative $\partial\log(P(t|c))/\partial P(t|c) = 1/P(t|c)$ seems promising in helping to interpret TF-IDF via disjoint terms (section 3.6.1).

3.5.5 Disjoint Terms: P(d|q,c)

The decomposition of $P(d|q,c)$ is analogous to equations 3.55.

$$P(d|q,c) = \sum_{t\in d} P(d|t)\cdot P(t|q,c) \tag{3.58}$$

$$= \frac{1}{P(q|c)}\cdot\sum_{t\in d} P(d|t,c)\cdot P(q|t,c)\cdot P(t|c) \tag{3.59}$$

$$=P(d|c)\cdot\sum_{t\in d} P(t|d,c)\cdot P(t|q,c)\cdot\frac{1}{P(t|c)} \tag{3.60}$$

The equation 3.60 has a similar explanation as for equation 3.57 in section 3.5.4. Although $P(d,q) = P(d)\cdot P(q|d) = P(q)\cdot P(d|q)$, and the decompositions are related, it is still interesting to investigate $P(d,q)$.

3.5.6 Disjoint Terms: P(d,q|c)

With the assumption that document "d" and query "q" are independent given term "t", formula 3.61 will be written as in formula 3.62.

$$P(d,q|c) = \sum_t P(d,q|t,c) \cdot P(t|c) \tag{3.61}$$

$$= \sum_t P(d|t,c) \cdot P(q|t,c) \cdot P(t|c) \tag{3.62}$$

Next, by replacing $P(d|t,c)$, $P(q|t,c)$ with $\frac{P(t|d) \cdot P(d|c)}{P(t|c)}$, $\frac{P(t|q) \cdot P(q|c)}{P(t|c)}$ respectively, and inserting the location based estimates into equation 3.62, we obtain:

$$P(d,q|c) = \frac{1}{N_L(c)} \cdot \sum_t n_L(t,d) \cdot n_L(t,q) \cdot \frac{1}{n_L(t,c)} \tag{3.63}$$

$P(d,q|c)$ equation: location frequencies only

Equation 3.63 contains location frequencies only. Therefore, the next step is to inject $P_D(t|c)$ to approach TF-IDF.

For approaching TF-IDF, the equation $n_L(t,c) = \frac{n_L(t,c)}{P_D(t|c)} \cdot P_D(t|c)$ injects $P_D(t|c)$. Since $\frac{n_L(t,c)}{P_D(t|c)} = N_D(c) \cdot avgtf(t,c)$, we obtain the following equation for the joint probability $P(d,q|c)$:

$$P(d,q|c) = \frac{\frac{1}{N_D(c)}}{N_L(c)} \cdot \sum_t \frac{n_L(t,d)}{avgtf(t,c)} \cdot n_L(t,q) \cdot \frac{1}{P_D(t|c)} \tag{3.64}$$

Equation 3.64 contains two components of TF-IDF: normalized within document term frequency $\frac{n_L(t,d)}{avgtf(t,c)}$, and inverse document frequency $\frac{1}{P_D(t|c)}$. $\frac{n_L(t,d)}{avgtf(t,c)}$ also is an interesting component, namely divergence from randomness:

$$\frac{n_L(t,d)}{avgtf(t,c)} \tag{3.65}$$

The rationale of this component is:

- a term with within-document frequency $n_L(t,d)$ *greater* than the expected frequency $avgtf(t,c)$, is a *good* term;
- a term with within-document frequency $n_L(t,d)$ *less* than expected, is a *poor* term.

The component $\frac{n_L(t,d)}{avgtf(t,c)}$ is *greater* than 1.0 for *good*, equal to 1.0 for *average*, and *less* than 1.0 for *poor* terms.

3.6 Document and Query Independence (DQI) Model

Whether a document and a query are independent can be a good indicator of the relevance between them. In this section we look into the DQI measurement, DQI based retrieval function and their relations to TF-IDF.

If document and query are independent, then $P(d,q|c) = P(d|c) \cdot P(q|c)$. This formula is different to formula 3.62, as for the latter one, the independent assumption is given term t.

If there is dependence between document and query, then $P(d,q|c)$ can be expressed by either $P(q|d,c) \cdot P(d|c)$ or $P(d|q,c) \cdot P(q|c)$. $P(q|d,c) \cdot P(d|c)$ is chosen as query likelihood is convenient to estimate. Again the estimation for $P(q|d,c)$ can be found in formula 3.56,

With dependence and independence assumption, DQI is derived as the follows:

$$\begin{aligned} DQI(d,q|c) &:= \frac{P(d,q|c)}{P(d|c) \cdot P(q|c)} = \\ &= \sum_t \frac{avgdl(c)}{avgtf(t,c)} \cdot \frac{n_L(t,d)}{N_L(d)} \cdot \frac{n_L(t,q)}{N_L(q)} \cdot \frac{1}{P_D(t|c)} \end{aligned} \tag{3.66}$$

This measure corresponds to the overlap of document and query:

- $DQI(d,q|c) > 1$: the overlap of document and query is *greater* than if they were dependent.
- $DQI(d,q|c) = 1$: document and query are conditionally independent.
- $DQI(d,q|c) < 1$, the overlap of document and query is *less* than if they were dependent.

The remaining question is: how does the *DQI* relate to TF-IDF? From equation 3.66, the following formulation of TF-IDF is born:

$$\begin{aligned} &RSV_{\text{DQI-TF-IDF}}(d,q,c) = \\ &= \sum_t \frac{avgdl(c)}{avgtf(t,c)} \cdot \frac{n_L(t,d)}{N_L(d)} \cdot \frac{n_L(t,q)}{N_L(q)} \cdot -\log P_D(t|c) && (3.67) \\ &= \sum_t \frac{avgdl(c)}{avgtf(t,c)} \cdot P_L(t|d) \cdot P_L(t|q) \cdot idf(t,c) && (3.68) \end{aligned}$$

Here, the inverse term probability (in 3.66) is replaced by the negative logarithm of the term probability. How can this be explained?

3.6.1 DQI and TF-IDF

TF-IDF can be interpreted as an area under the DQI curve. This is because of the following integral:

$$\int \frac{1}{x}\,\mathrm{d}x = \log x \tag{3.69}$$

Through this, the inverse term probability $\frac{1}{P_D(t|c)}$ can be related to the inverse document frequency $idf(t,c) = -\log P_D(t|c)$. This is achieved by the definite integral ranging from $P_D(t|c)$ to 1.0.

$$\int_{P_D(t|c)}^{1.0} \frac{1}{x}\,\mathrm{d}x = \log(1.0) - \log P_D(t|c) = idf(t,c) \tag{3.70}$$

Figure 3.1 illustrates this interpretation of TF-IDF.

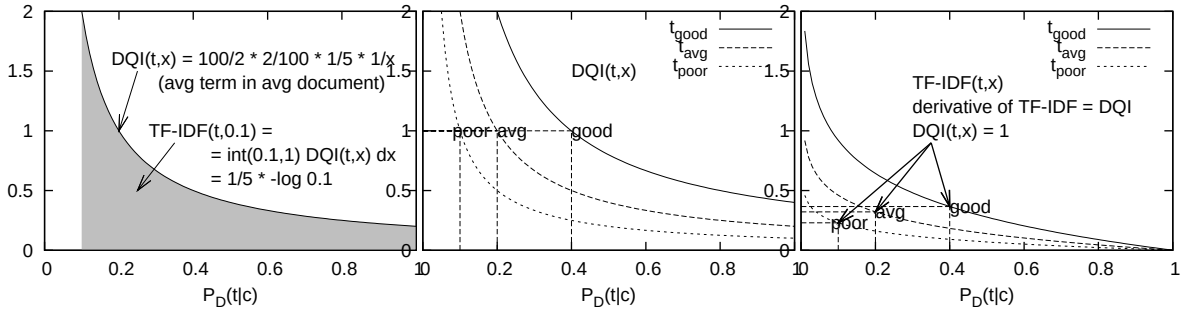


Figure 3.1: TF-IDF is integral of Document-Query Independence (DQI) over term probability $\mathbf{P_D(t|c)}$

The left plot shows TF-IDF to be the area under the *DQI* curve. The y-axis shows the *DQI* values for an average term ($n_L(t,d) = avgtf(t,c)$) in an average document ($N_L(d) = avgdl(c)$), and the x-axis corresponds to the probability $P_D(t|c)$. The example shows a term with $n_L(t,d) = 2$, in a document with $N_L(d) = 100$. The query has five terms, i.e. $P_L(t|q) = 1/5$.

The middle plot shows *DQI* curves for three terms: an *average* term, a *good* term ($n_L(t_{\text{good}},d) = 2 \cdot avgtf(t_{\text{good}},c)$), and a *poor* term ($n_L(t_{\text{poor}},d) = 0.5 \cdot avgtf(t_{\text{poor}},c)$). For example, the curve for t_{avg} follows from $avgdl(c)/avgtf(t,c) \cdot P_L(t_{\text{avg}}|d) = 1$. Then, $P_L(t_{\text{avg}}|q) \cdot 1/x$ is the *DQI* of t_{avg}, where x is the probability $P_D(t|c)$ that term t occurs in a document of collection c. For example, if $P_D(t_{\text{avg}}|c) = 0.2$, then $DQI(t_{\text{avg}}, 0.2) = 1$.

The right plot shows the TF-IDF values of the three terms, i.e. the TF-IDF values correspond to the area under the respective *DQI* curve in the middle plot. In the right plot, the TF-IDF values for a good term at $P_D(t|c) = 0.4$, an average term at $P_D(t|c) = 0.2$, and a poor term at $P_D(t|c) = 0.1$ are marked, since at these points, the gradient of the TF-IDF curve is equal to 1.0, i.e. this is the

probability $P_D(t|c)$ where the TF-IDF slope goes from large to small.

For facilitating the mathematical expressions to follow, the inverse average term frequency *iatf* of a term is defined as follows:

$$\text{inverse average term frequency:} \quad iatf(t,c) := \frac{avgdl(c)}{avgtf(t,c)} \tag{3.71}$$

For example, in a collection with $avgdl(c) = 1,000$, if a term occurs on average 10 times in its elite set ($avgtf(t,c) = 10$), then $iatf(t,c) = 1,000/10 = 100$.

When DQI and DQI based TF-IDF are defined as follows,

$$DQI(t,x) = iatf(t,c) \cdot P_L(t|d) \cdot P_L(t|q) \cdot \frac{1}{x} \tag{3.72}$$

$$RSV_{DQI\text{-TF-IDF}}(t) = iatf(t,c) \cdot P_L(t|d) \cdot P_L(t|q) \cdot idf(t,c)x \tag{3.73}$$

Then TF-IDF an be interpreted as as the integral of *DQI* over the probability $x = P_D(t|c)$:

$$RSV_{DQI\text{-TF-IDF}}(t) = \int_{P_D(t|c)}^{1.0} DQI(t,x)\,\mathrm{d}x \tag{3.74}$$

The values of the occurrence probability $x = P_D(t|c)$ for which $DQI(t,x) = 1$ holds, are deemed to be of particular interest. Therefore, the DQI and TF-IDF plots show the respective points. These are where TF-IDF changes from fast to slow fall. The *DQI* threshold $DQI(t, P_D(t|c)) = 1$ seems to open new opportunities for judging the power of terms to discriminate between relevant and non-relevant documents.

3.6.2 DQI and Mutual Entropy

Mutual entropy is widely used in crossing language. [Church and Hanks, 1990, Gale and Church, 1991] have used mutual entropy between the terms $\log \frac{p(t_i,t_j)}{P(t_i)P(t_j)}$ to study the the word association in the same corpus or term dependence.

In DQI model, query and document are viewed as two undivided unit respectively. If query and document are decomposed to term level, the mutual information of document and query I(d:q) can be defined as follows:

$$I(d:q) = \sum_{t_j \in d} \sum_{t_j \in q} P(t_i,t_j) \log \frac{P(t_i,t_j)}{P(t_i)P(t_j)} \tag{3.75}$$

$P(d_i,q_i)$ can be estimated based on co-occurrence of the terms.

DQI measure is based on the independence of query and document, whereas I(d:q) is based the independence query and document on term level.

3.7 Summary

The motivation for this chapter is to clarify the mathematical formalism of probabilistic models, theoretically justify TF-IDF model. We parallelly investigate the three grand probabilistic retrieval models: BIR, PM, and LM, from the event spaces, background models, frequencies, probabilities, term probability interpretations, parameter estimations to retrieval status values.

The parallel investigation of the models showed: PM can be viewed as a bridge connecting BIR and LM, thus PM can explain TF-IDF with either BIR or LM probabilities in a dual way.

The interpretations of TF-IDF were also explored systematically by decomposing relevance probability based on independent and disjoint terms.

For independent terms, the LM-like decomposition of $P(d|q,c)$ yields an interpretation of TF-IDF, with an extreme query/non-query assumption: $P(t|q,c) = P(t|q)$ for query terms, and $P(t|q,c) = P(t|c)$ for non-query terms. Decomposition of probabilistic odds leads to a TF-IDF interpretation showing an analogy between $P(d|q,c)$ and $O(r|d,q)$; this backs the approximations of term probabilities expressed in the equations $\overline{r} = c$ and $r = q$.

For disjoint terms, the decomposition of $P(d,q|c)$ yields manifold results. Divergence from randomness and pivoted document length are shown to be inherent parts of a document-query independence (DQI) measure, where the DQI follows the application of maximum-likelihood estimations. The interpretation of TF-IDF as integral of the DQI uncovers novel meanings and properties of TF-IDF.

Chapter 4

Implementing Retrieval Models with High Abstract Languages

In the past half century, many kinds of retrieval systems have been developed. In the early stages, the file representations were managed in the file system. With the development of relational databases, some IR systems store the file representations in the database to utilize the powerful data management functions of large relational database manage system. Their retrieval models, the matching and ranking of the document to the query, are mostly implemented with structured programming languages such as C++, Java or Perl etc. These implementations are hard to prove whether they express the mathematical models in exactly the same way and hard to trace mistakes generated during the implementation.

In the last ten years, IR models have attracted great interests from database researchers, who used IR models to rank database retrieval results [Agrawal et al., 2003, Cohen, 1998, Chaudhuri et al., 2006, Chandel et al., 2007]. Some of them integrate the ranking functions into the database system, which loses the flexibility in term of ranking strategy. Others use standard SQL, but have to explicitly dealing with probability computation.

The integration of DB and IR, probabilistic database, possess the capability for modelling the uncertainty of knowledge, and free users from the direct probability manipulations. The motivation of probabilistic relational modelling for us is to provide an approach for implementing retrieval strategies, which is possible to describe and rank a well-defined ranking of objects in a relational database. The objects are not restricted to texts, can be extended to projects, persons,

products, etc. On the other hand, it is also to provide the ability to easily modify the retrieval model.

In this chapter we present how some famous contemporary IR models are implemented in High Abstract Languages - Probabilistic Relational Algebra and Probabilistic SQL. Before starting the modelling, we give a brief introduction to the languages of probabilistic relational algebra and probabilistic SQL.

4.1 High Abstract Languages

The platform that we use is HySpirit [Fuhr and Roelleke, 1997], which is a probabilistic retrieval framework with four abstraction layers: probabilistic object oriented logic (POOL), four-valued probabilistic Datalog (FVPD), probabilistic Datalog (PD) and probabilistic relational algebra (PRA). It provides concise syntax and powerful probabilistic relational modelling ability. Recently, another user friendly interface, Probabilistic SQL (PSQL), has been developed. PSQL has a similar syntax to standard SQL, whilst including probability estimation and aggregation capabilities. PSQL needs to be explained as PRA for executed, which means that any PSQL statement has an equivalent PRA expression. Depending on the actual preference, the user can choose any appropriate abstraction layer to build their retrieval models.

Our modelling work is mainly based on PRA level, at the mean time we implement the models with PSQL to make it more readable to non-computer scientist. Formal probability estimation and aggregation definitions can also be found in [Roelleke et al., 2008]. Here we give the syntax of PRA and PSQL and some examples to help in understanding the later sections' retrieval model implementations.

4.1.1 Probabilistic Relation Algebra (PRA)

PRA comes from relation algebra and probability theory. It is the lowest level of HySpirit, and all the other level languages need to be translated into PRA in order to be executed.

Apart from 5 basic relational operators (*PROJECT*, *SELECT*, *JOIN*, *UNITE*, *SUBTRACT*), there is one extra probability estimation operator *BAYES* included in PRA. The *BAYES* operator solves the problem of probability estimation from non-probabilistic data, integration probability estimation into the retrieval model. The syntax of the operators is as shown in Table 4.1.

PROJECT::= '*PROJECT*' assumption [target_list] (expression)
SELECT::= '*SELECT*' [cond_list] (expression)
JOIN::= '*JOIN*' assumption [cond_list] (expression,expression)
UNITE::= '*UNITE*' (expression,expression)
SUBTRACT::= '*SUBTRACT*' assumption (expression,expression)
BAYES::= '*BAYES*' assumption [evidence key] (expression)

Table 4.1: Basic operators of PRA

The first 5 operators are almost the same as traditional relational algebra, whilst they provide probability aggregation for each tuple. The assumption option for the operator includes the probability assumption, which can be *disjoint*, *independent* or *subsume*, also the assumptions in the relational operation which may also be beyond classical probability theory and aims to facilitate IR model modelling. For different operators, they might have different assumptions. Table 4.2 lists all the assumptions for each operator:

Operator	Assumption
PROJECT	disjoint, independent, subsumed, distinct, sum_log, max_log
JOIN	mixed, exponential
UNITE	disjoint, independent, subsumed
SUBTRACT	disjoint, independent, subsumed
BAYES	sum, max, sum_idf, max_idf

Table 4.2: Assumptions for each operator

The evidence key is a list of column (attribute) identifiers, it is used to specify the condition of the probability estimation.

Next we will give a definition of each of PRA operator, with examples of how it works based on one mini-collection and one query in table 4.4.

Before we start the definition, we describe all of symbols used in this section:

Notation	Explanation
(T,P)	is a probabilistic relation, which consists of a tuple set T and a probability set P. T and P have the same set size, each probability value corresponds a tuple.
τ	is a tuple in a relation, which contain a few attributes $(\tau_1, \tau_2 ... \tau_n)$.
i_n	is the index of an attribute in a tuple. Therefore $\tau[i_,i_2,...i_n]$ is a new tuple which contains of the attributes $i_1, i_2, ...i_n$ from tuple τ.
$P(\tau)$	returns the probability of tuple τ.

Table 4.3: Notations in the definitions of PRA operators

Suppose the mini-collection *coll* only contains the occurrence information, lets see the result after each operation.

coll		
	Term	DocId
1	sailing	doc1
1	sailing	doc1
1	sailing	doc1
1	boats	doc1
1	boats	doc1
1	fish	doc2
1	boats	doc2
1	segull	doc3
1	boats	doc3
1	sailing	doc3

query		
	QTerm	QueryId
1	boats	q1
1	fish	q1

Table 4.4: Representation of collection and query

Relational BAYES

The relational *BAYES* operator is a special operator for estimating frequency-based probability e.g. probability of term occurring in a document (tf = *PROJECT* [$Term,$DocID](*BAYES* [$DocID](coll))). It can also be used to compute term informativeness (IR feature), e.g. inverse document frequency (idf = *BAYES* max_idf [ ] (*PROJECT* [$Term] (Coll))), which is tailored specially to IR tasks. The max_idf is used to normalize the IDF value with the maximum IDF from the collection.

Definition 1 ***BAYES*:**

$(T,P) =$ *'BAYES' assumption*$[i_1 \ldots i_n]$*(a)*

$$\begin{aligned} T &:= \{\tau | \tau \in T_a\} \\ P(\tau) &:= \frac{P_a(\tau)}{P_b(\tau[i_1 \ldots i_n])} \end{aligned}$$

The key $i_1 \ldots i_n$ *is referred to the* evidence key *since the relational BAYES generates a relation where the tuple probabilities correspond to the conditional probability* $P(\tau|\tau[i_1 \ldots i_n])$.

The probabilistic relation "b" is the so-called evidence key *projection:*

b = 'PROJECT' assumption$[i_1 \ldots i_n]$*(a).*

If no assumption is specified, i.e. given BAYES[...](...), then the assumption 'disjoint' is the default.

$\diamond$ *end of definition*

Table 4.5 shows two probabilistic relations *tf_sum* and *idf*, which are computed from *coll* with *BAYES* operator. To compute *tf*, the evidence key is document ID. And evidence key is empty for computing *idf*, as the *idf* is estimated based on the whole collection.

tf		
$P(t\|d)$ (*tf_sum*)	Term	DocId
0.600000	sailing	doc1
0.400000	boats	doc1
0.500000	fish	doc2
0.500000	boats	doc2
0.333333	segull	doc3
0.333333	boats	doc3
0.333333	sailing	doc3

idf	
P(idf)	Term
0.369070	sailing
0.000000	boats
1.000000	fish
1.000000	segull

PROJECT sum [$1,$2] (*BAYES* [$2](Coll)) *BAYES* max_idf[] (*PROJECT* [$1] (*coll*))

Table 4.5: Operations based on *BAYES*

PROJECT

PROJECT is an operator used for probability aggregation based on particular columns of a relation, it can also select distinct tuples from the relation.

Definition 2 *PROJECT*:

Let $\tau = \tau'[i_1..i_n]$ be a tuple composed of the attribute values at columns (positions) $i_1..i_n$ in tuple τ', and T_a be the set of tuples of relation "a" that share the same attribute values at columns $i_1..i_m$.

(T,P) = 'PROJECT' assumption[praTargetList](a)

$$
\begin{aligned}
T &:= \{\tau | \tau = \tau'[i_1..i_n] \wedge \tau' \in T_a\} \\
P(\tau) &:= \begin{cases}
\sum_{\tau' \in T_a(i_1..i_n)} P_a(\tau') \text{ if assumption='disjoint'} \\
1 - \prod_{\tau' \in T_a(i_1..i_n)} (1 - P_a(\tau'[i_1..i_n])) \\
\qquad \text{if assumption='independent'} \\
\max(\{P_a(\tau') | \tau'[i_1..i_n] \in T_a(i_1..i_n)\} \\
\qquad \text{if assumption='subsumed'}
\end{cases}
\end{aligned}
$$

Apart from the above three basic probabilistic assumptions, there are two supplementary assumptions for convenience in IR modelling: complement and sum_log. The Complement assumption can be used to compute $P(\bar{a})$ when $P(a)$ already exists; while the sum_log is used for

the product of the probability from the tuples in the same relation, e.g. the probability of the terms sailing *and* boats *occurring in the document* doc1 *at the same time based on independent assumption.*

$$P(\tau) := \begin{cases} 1 - P_a(\tau) & \textit{if assumption='complement'}; \\ \prod_{\tau[i_1..i_n] \in T_a(i_1..i_n)} P_a(\tau) & \textit{if assumption='sum-log'}. \end{cases}$$

If no praTargetList is specified, i.e. PROJECT assumption(a), then this is equivalent to the praTargetList that contains all attributes of the argument relation "a".

◇ *end of definition*

Table 4.6 shows some results from *PROJECT* based on distinct, disjoint and independent assumptions respectively.[1]

doc	
	DocId
1.000000	doc1
1.000000	doc2
1.000000	doc3

PROJECT distinct [$2](*coll*)

dl	
	DocId
5.000000	doc1
2.000000	doc2
3.000000	doc3

PROJECT disjoint [$2](*coll*)

weighted_doc	
	DocId
0.760000	doc1
0.750000	doc2
0.703704	doc3

PROJECT independent [$2](*tf*_sum)

Table 4.6: *PROJECT* with different assumptions

SELECT

SELECT is an operator used for choosing the tuples that satisfy a certain condition.

Definition 3 *SELECT*:

$(T,P) =$ *'SELECT'[condition](a)*

$$\begin{aligned} T &:= \{\tau | \tau \in T_a \wedge \varphi(\tau)\} \\ P(\tau) &:= P_a(\tau) \end{aligned}$$

Here, φ represents the semantic truth value function that corresponds to the syntactic "condition" in the selection.

◇ *end of definition*

[1]The value of for each document length in here is greater than 1, which is not probabilistic. This can be solved by normalizing the term with collection length (*PROJECT* [$2] (*BAYES* [](*coll*))). Here we are only to show the probability can be summed up with *PROJECT*operator.

For example we want to show all the terms in *doc*1, we use *SELECT* [$2=doc1](*coll*) (see Table 4.7).

*doc*1		
	Term	DocId
1.000000	sailing	doc1
1.000000	sailing	doc1
1.000000	sailing	doc1
1.000000	boats	doc1
1.000000	boats	doc1

Table 4.7: *SELECT*: *doc*1=*SELECT* [$2 = doc1](*coll*)

JOIN

JOIN is used to connect the two relations, which can be a conditional join or just a cross product when no condition presented. The probabilities from two relations will be multiplied to form the new probability of the tuple in the result relation.

Definition 4 *JOIN*:

$(T,P) =$ *'JOIN' assumption(a,b)*

$$
\begin{aligned}
T &:= \{\tau | \tau_a \in T_a \wedge \tau_b \in T_b \wedge \tau = [\tau_a, \tau_b]\} \\
P(\tau) &:= \begin{cases} 0 \text{ if assumption='disjoint'} \\ P_a(\tau_a) \cdot P_b(\tau_b) \\ \qquad \text{if assumption='independent'} \\ \min(\{P_a(\tau_a), P_b(\tau_b)\}) \\ \qquad \text{if assumption='subsumed'} \end{cases}
\end{aligned}
$$

$\diamond$ *end of definition*

Table 4.8 shows the result when relations *coll* (in Table 4.4) and *idf* in Table 4.5 are joined together with condition that terms from each relation are the same, each term in the collection *coll* has a *idf* weight:

UNITE and SUBTRACT

UNITE, SUBTRACT are used to merge two relation, and aggregate the probability. If the aggregated probability is less than zero, then the tuple probability will be set to zero.

coll_weighted			
	Term	DocId	Qterm
0.369070	sailing	doc1	sailing
0.369070	sailing	doc1	sailing
0.369070	sailing	doc1	sailing
0.000000	boats	doc1	boats
0.000000	boats	doc1	boats
1.000000	fish	doc2	fish
0.000000	boats	doc2	boats
1.000000	segull	doc3	segull
0.000000	boats	doc3	boats
0.369070	sailing	doc3	sailing

Table 4.8: *JOIN*: coll_weighted = JOIN [$1=$1](Coll, p_idf)

Definition 5 ***UNITE*:**

$(T,P) =$ *'UNITE' assumption(a,b)*

$$
\begin{aligned}
T &:= \{\tau | \tau \in T_a \vee \tau \in T_b\} \\
P(\tau) &:= \begin{cases}
P_a(\tau) + P_b(\tau) \text{ if assumption='disjoint'} \\
P_a(\tau) + P_b(\tau) - P_a(\tau) \cdot P_b(\tau) \\
\qquad \text{if assumption='independent'} \\
\max(\{P_a(\tau), P_b(\tau)\}) \\
\qquad \text{if assumption='subsumed'}
\end{cases}
\end{aligned}
$$

$\diamond$ *end of definition*

Definition 6 ***SUBTRACT*:**

$(T,P) =$ *'SUBTRACT' assumption(a,b)*

$$
\begin{aligned}
T &:= \{\tau | \tau \in T_a\} \\
P(\tau) &:= \begin{cases}
P_a(\tau) \text{ if assumption='disjoint'} \\
P_a(\tau) \cdot (1 - P_b(\tau)) \\
\qquad \text{if assumption='independent'} \\
P_a(\tau) - P_b(\tau) \\
\qquad \text{if assumption='subsumed'}
\end{cases}
\end{aligned}
$$

$\diamond$ *end of definition*

To illustrate how the assumption affect the result, we give two relations that have identical attributes in table 4.9, then show the results of unite and subtract with different assumptions in table 4.10 and table 4.11.

idf	
	Term
0.369070	sailing
0.000000	boats
1.000000	fish
1.000000	segull

*idf*1	
	Term
0.7	sailing
0.3	panda

Table 4.9: Two relations with the same attributes

idf	
	Term
1.069070	sailing
0.000000	boats
1.000000	fish
1.000000	segull
0.300000	panda

UNITE disjoint(*idf*,*idf*1)

idf	
	Term
0.810721	sailing
0.000000	boats
1.000000	fish
1.000000	segull
0.300000	panda

UNITE independent (*idf*,*idf*1)

idf	
	Term
0.700000	sailing
0.000000	boats
1.000000	fish
1.000000	seagull
0.300000	panda

UNITE subsumed (*idf*,*idf*1)

Table 4.10: *UNITE* with different assumptions

4.1.2 Probabilistic SQL (PSQL)

PSQL has the same syntax as standard SQL apart from a few new features concerned with probability estimation and aggregation during query processing. Other data definition and manipulation statements (i.e. *CREATE*, *INSERT*, *DROP*) have no difference in terms of both syntax and operation. Therefore, we are not going to introduce those statements. Readers who are interested in SQL can refer relevant book, note that our PSQL does not support transaction or consistency validation the complex function. It can create table, view, create index, insert data, remove data or drop table. *UNION* and *MINUS* have to be used in conjunction with select to perform set minus or sum. It is inevitable to do probability aggregation for the tuples. The probability assumption are the same as *UNITE* and *SUBTRACT* that have introduced in section 4.1.1. We only describe *SELECT* in this section

The syntax of *SELECT* statement is as follows:

```
psqlSelect ::='SELECT' sqlTargetList
    'FROM' relationList
```

idf	
	Term
0.369070	sailing
0.000000	boats
1.000000	fish
1.000000	segull

SUBTRACT disjoint (*idf*,*idf* 1)

idf	
	Term
0.110721	sailing
0.000000	boats
1.000000	fish
1.000000	segull

SUBTRACT independent (*idf*,*idf* 1)

idf	
	Term
0.000000	sailing
0.000000	boats
1.000000	fish
1.000000	seagull

SUBTRACT subsumed (*idf*,*idf* 1)

Table 4.11: *SUBTRACT* with different assumptions

```
['WHERE' sqlCondition]
[aggAssumption]['EVIDENCE KEY' (sqlTargetList) ]
```

- aggAssumption ::= 'ASSUMPTION' assumption
- assumption ::= 'disjoint' | 'independent' | 'subsumed'
- sqlTargetList ::= ... *as in SQL* ...
- relationList ::= ... *as in SQL* ...
- sqlCondition ::= ... *as in SQL* ...

'*SELECT*...*FROM*...*WHERE*...' statement in PSQL is executed the same as standard SQL, apart from it probability aggregation and estimation function. All the probability aggregation assumptions applied to PRA are adopted in PSQL too. EVIDENCE KEY sub-clause indicates the *BAYES* operation in the query statement. Detailed information about probability aggregation assumption and *BAYES* operation can be find in section 4.1.1.

4.2 Probabilistic Relational Modelling of Retrieval Models

4.2.1 Simple Modelling Example

In this section we give an example of modelling a basic retrieval model, assuming we have a relational representation of a TF-based document index, an IDF-based term space, and a query (see Table 4.12).

Given such a knowledge representation, TF-IDF retrieval retrieval strategy can be described in PSQL (Probabilistic SQL) and PRA (Probabilistic Relational Algebra) as follows. Note that PSQL and PRA's comments start with '−−' and '##', respectively.

tf		
$P(t\|d)$	Term	DocId
0.5	sailing	doc1
0.5	boats	doc1
0.6	sailing	doc2
0.4	boats	doc2

idf		
$P(t\|c)$	Term	Collection
0.1	sailing	c1
0.8	boats	c1

query	
Term	QueryId
sailing	q1
boats	q1

Table 4.12: Representations of TF, IDF and Query

```
1   -- IDF-based query term weighting:
2   CREATE VIEW weightedQuery AS
3     SELECT ALL Term, QueryId
4     FROM query, idf
5     WHERE query.Term = idf.Term;

7   -- TF-IDF-based retrieval:
8   CREATE VIEW retrieve AS
9     SELECT DISJOINT Term, QueryId
10    FROM weightedQuery, tf
11    WHERE weightedQuery.term = tf.Term;
```

The above PSQL program is equivalent to the PRA program as follows:

```
1   # IDF-based query term weighting:
2   weightedQuery = PROJECT ALL[$1,$2](JOIN[$1=$1](query, idf));

4   # TF-IDF-based retrieval:
5   retrieve  = PROJECT DISJOINT[$4,$2](JOIN[$1=$1](weightedQuery, tf));
```

For the relations (views) "weightedQuery" and "retrieve", we obtain them in Table 4.13:

weightedQuery		
Prob	Term	QueryId
0.10	sailing	q1
0.80	boats	q1

retrieve		
Prob	DocId	QueryId
0.45	doc1	q1
0.38	doc2	q1

Table 4.13: Weighted query and retrieval result

For example, $P_{\text{retrieve}}(doc1, q1) = 0.45$ is the result of $0.1 \cdot 0.5 + 0.8 \cdot 0.5$, where $P_{\text{weightedQuery}}(sailing, c1) = 0.1$, and $P_{\text{tf}}(sailing, doc1) = 0.5$, and so forth. The *JOIN* over the

terms leads to the multiplication of probabilities, and the disjoint *PROJECT* adds the query term weights together to form the document retrieval status value.

Next, we focus on the probabilistic relational modelling of the models. For the purposes of this chapter, we restrict them to the classical case of document retrieval only.

4.2.2 TF-IDF Modelling

The standard definition of the TF-IDF-based retrieval status value (RSV) is of the form $RSV(d,q) = \sum_{t \in d \cap q} tf(t,d) \cdot idf(t)$. When investigating the implementation of TF-IDF in a probabilistic relational framework, we came across different variants which we will report in this section. For implementing the standard form, we need to instantiate probabilistic relations to model TF and IDF. Since we move in a probabilistic framework, we need to think about a probabilistic interpretation of TF-IDF, or, at least, define probabilities that are proportional to TF and IDF respectively. This is fairly straight-forward for the TF component, but for the IDF component, we need a log-based normalization and the probabilistic interpretation of the value obtained is not obvious (see [Roelleke, 2003b] for a discussion of the semantics of such a probability).

We illustrate this in the following several TF-IDF implementations. One is the standard TF-IDF, and one is a simple alternative but with TF-IDF features.

First, lets look at the PSQL script for modelling standard TF-IDF-based retrieval. There are views for defining the probabilistic relations "*tf*" and "*idf*", which is the merit of probabilistic modelling mentioned before. The probability estimation and retrieval strategy modelling can be integrated in a few lines, once you have the representation of the document collection.

```
1  -- PSQL: standard TF-IDF retrieval
2  -- Extensional relations :
3  -- coll(Term, DocId);  query(Term, QueryId);  tf_poissona (Term, DocId);

5  -- within-document term frequency:
6  CREATE VIEW tfCollSpace AS
7          SELECT Term, DocId
8          FROM coll
9          ASSUMPTION DISJOINT
10         EVIDENCE KEY (DocId);
11 CREATE VIEW tf AS
12         SELECT DISJOINT Term, DocId
13         FROM tfCollSpace;

15 -- Optional: Bind tf to extensional relation .
16 --CREATE VIEW tf AS
```

```
17 --        SELECT Term, DocId
18 --        FROM tf_poissona;

20 -- inverse document frequency:
21 CREATE VIEW idf AS
22         SELECT Term
23         FROM coll
24         ASSUMPTION MAX_IDF
25         EVIDENCE KEY ();

27 -- query term weighting and normalization :
28 CREATE VIEW wQuery AS
29         SELECT Term, QueryId
30         FROM query, idf
31         WHERE query.Term = idf.Term;
32 CREATE VIEW norm_wQuery AS
33         SELECT Term, QueryId
34         FROM wQuery
35         EVIDENCE KEY (QueryId);

37 -- retrieve documents:
38 CREATE VIEW std_tf_idf_retrieve AS
39         SELECT DISJOINT DocId, QueryId
40         FROM norm_wQuery, tf
41         WHERE norm_wQuery.Term = tf.Term;

43 CREATE VIEW retrieve AS
44         SELECT DocId, QueryId
45         FROM  std_tf_idf_retrieve ;
```

The PSQL script contains views for defining the probabilistic relations "*tf*" and "*idf*". For "*tf*", the first two views demonstrate how to define a maximum-likelihood estimate, which is of the form $P(t|d) = n(t,d)/N(d)$. This linear estimate is outperformed by a non-linear estimate of the form $n(t,d)/(n(t,d)+K)$, where $n(t,d)$ is the number of times term t occurs in document d, and K is a term-independent value, which might reflect, for example, the document length (BM25, [Robertson et al., 1995]). This non-linear estimate can be viewed as a Poisson approximation, and the term-document pairs with the respective probabilities are stored in relation "*tf_poissona*".

The query terms are joined with "*idf*" to generate the relation "*wQuery*" of weighted query terms. The normalized query terms are required for obtaining a probabilistic interpretation of the sum over the TF-IDF products. Finally, we define the view "*std_tf_idf_retrieve*", which contains the document-query pairs with their probabilistic TF-IDF retrieval status values.

The translation of the PSQL script yields an equivalent PRA program which is shown below.

```
1  # PRA: TF−IDF retrieval
2  # Extensional relations :
3  # coll (Term, DocId); query(Term, QueryId); tf_poissona (Term, DocId);

5  # tfCollSpace (Term, DocId):
6  tfCollSpace = BAYES[$2](coll);
7  # tf (Term, DocId):
8  tf = PROJECT disjoint[$1,$2](tfCollSpace);

10 # Optional: Bind tf to extensional relation .
11 #tf = tf_poissona ;

13 # idf (Term):
14 idf = BAYES max_idf[](PROJECT[$1](coll));

16 # wQuery(Term, QueryId):
17 wQuery = PROJECT[$1,$2](JOIN[$1=$1](query, idf));

19 # Normalization:
20 norm_wQuery = PROJECT[$1,$2](BAYES[$2](wQuery));

22 # Retrieve documents:
23 # std_tf_idf_retrieve (DocId, QueryId):
24   std_tf_idf_retrieve = PROJECT disjoint[$4,$2](JOIN[$1=$1](norm_wQuery, tf));

26 retrieve = std_tf_idf_retrieve ;
```

Each PRA equation corresponds to a view in the PSQL script. PSQL views that involve evidence keys or assumptions lead to PRA expressions in which the relational *BAYES* performs the required probability estimation. This is the case for the view "*tfCollSpace*", and for the view "*idf*".

We have modeled standard TF-IDF. The maximum-likelihood estimation is a conceptual part of the minimal probabilistic relational framework we have presented so far. It is one of the main contributions of the relational *BAYES* that such estimations are now part of the probabilistic relational paradigm, and do not need to be computed *outside* of the relational algebra, and do not need the instantiated TF relation. For non-linear estimation, an approximation of the 2-Poisson process, we still bind "*tf*" to the extensional relation "*tf_poissona*" in which probabilities were generated offline. There are several ways in the PSQL/PRA framework to compute 2-Poisson approximated probabilities, however, our aim is to integrate probability estimations neatly into the conceptual framework of probabilistic relational modelling, rather than to invent new assumptions or SQL syntax extensions for various probabilities estimation. In this chapter we focus on

the minimal PRA and its relational *BAYES*.

When implementing TF-IDF, we encountered less complex PSQL programs that provide a TF-IDF-like RSV. Consider the following alternative and fairly compact PSQL program, where we join IDF-weighted query terms with the relation "*coll*" rather than "*tf*". In "*coll*", we have non-distinct Term-DocId tuples, whereas in "*tf*", tuples are distinct since the non-distinct Term-DocId tuples have been aggregated into the probabilities of the tuples in "*tf*".

```
1  -- PSQL: alternative TF-IDF-like retrieval
2  -- This TF-IDF variant does not rely on the generation of an explicit tf relation.

4  CREATE VIEW alt1_tf_idf_retrieve AS
5      SELECT INDEPENDENT DocId, QueryId
6      FROM wQuery, coll
7      WHERE wQuery.Term = coll.Term;
```

Corresponding PRA:

```
1  alt1_tf_idf_retrieve = PROJECT independent[$4,$2]( JOIN[$1=$1](wQuery, coll));
```

The independent assumption leads to an aggregation of the query term probabilities that we obtain from the probabilities in "*alt1_tf_idf_retrieve*": $RSV(d,q) = 1 - \prod_{(t,d) \in Coll}(1 - P(q|t))$. Note that the aggregation of non-distinct (t,d) tuples in the relation "*coll*" reflects the within-document term frequency. The light-weight nature of this implementation motivated us to investigate other similar forms against TF-IDF-implementations that contain an explicit relation "*tf*".

For another candidate with explicit "*tf*" relation, consider the following script in which we join the non-normalized rather than the normalized query term weights, and view the query terms as independent rather than disjoint, we then obtain another implementation of TF-IDF:

```
1  --PSQL: alternative TF-IDF-like retrieval
2  -- Aggregation of independent, non-normalized query term weights.

4  CREATE VIEW alt2_tf_idf_retrieve AS
5      SELECT INDEPENDENT DocId, QueryId
6      FROM wQuery, tf
7      WHERE wQuery.Term = tf.Term;
```

Corresponding PRA is:

```
1  alt12_tf_idf_retrieve = PROJECT independent[$4,$2]( JOIN[$1=$1](wQuery, tf));
```

Note the difference between "*alt2_tf_idf_retrieve*" and "*std_tf_idf_retrieve*": In

"*alt2_tf_idf_retrieve*", we (have to) apply an independence assumption. In "*std_tf_idf_retrieve*", we (had to) normalize the weighted query terms for the safe application of a disjoint projection.

It is not our aim in this chapter to study the performance of the retrieval model. However, the variants of TF-IDF that emerged when modelling TF-IDF in PSQL/PRA intrigued us enough to investigate their performances. We ran the TF-IDF variants on the 500MB structured INEX collection with around 12K articles , 15 million retrievable contexts (sections, paragraphs, etc), and 32.5 million terms. The representation of INEX is identical to the relation "*coll*" in our running example. Due to the fact that the INEX test collection is designed for element retrieval and assessment[Fuhr et al., 2003a], it assess the element of a document from two aspects specificity and exaustivity with scale 0 to 3. We adapted the element assessments to document assessment, where the rule is:

> If an element has any aspect judged with value greater than 0 ($specificity > 0$ or $exaustivity > 0$), then this element is relevant.
>
> If any element of a document is relevant, then this document is relevant.

Here, we use mean average precision (MAP) and precision at 10 retrieved document (P@10) to indicate the retrieval quality. For these TF-IDF variants, we obtain the retrieval quality presented in table 4.14, where the variants are sorted by performance.

TF-IDF	tf	wQuery	MAP	P@10
std1	Poisson *tf*	normalized	0.2713	0.4138
std2	Likelihood *tf*	normalized	0.2077	0.4103
alt1	implicit *tf*	non-normalized	0.2038	0.4091
alt2	Likelihood *tf*	non-normalized	0.1224	0.2586

Table 4.14: Retrieval quality for TF-IDF alternatives

The experiment confirms that TF-IDF with Poisson-approximated TF performs best. The standard variants (std1 and std2) work with normalized IDF-based probabilities for query term weighting, whereas the alternative variants (alt1 and alt2) work with non-normalized query term weights. The variant with implicit TF, where the join of query terms with the relation "*coll*" followed by an independent projection implicitly captures the TF part, performs quite well, taking into account that this implementation actually frees the system from providing a view "*tf*" or even a materialized relation.

The aim of this part is to demonstrate that PSQL/PRA are flexible with respect to retrieval

strategy modelling, capable in queries involving complex relation schema, and suitable for large scale data. They also provide various methods in probability estimation and aggregation. Above all they allow us to formulate and investigate retrieval models in an abstract, relatively compact representation.

In later sections we will use PRA/PSQL to implement some famous contemporary probabilistic models.

4.2.3 Binary Independence Retrieval Model

In the BIR model, there are F1 - F4 term weights (see section 2.4.2), we implement F1 weight, and other weights can be implemented in the same way. F1 weight is:

$$w_{F_1} = \log \frac{P_D(t|r)}{P_D(t|c)} = \log \frac{n_D(t,r)/N_D(r)}{n_D(t,c)/N_D(c)} = \log \frac{n_D(t,r)}{N_D(r)} - \log \frac{n_D(t,c)}{N_D(c)} \tag{4.1}$$

After reforming the the equation 4.1 with the definition of IDF, we obtain the F1 weight as $w_{F_1} = idf(t,c) - idf(t,r)$ which is central to the BIR model implementation.

The PRA program contains the equations (views) for implementing the probabilistic variants of the BIR model. The PSQL program is equivalent to the PRA program .

```
1  # Extensional relations :
2  # coll (Term, DocId); query(Term, QueryId); relevant (QueryId, DocId);

4  # Part 1: Basic declarations

6  # queries (QueryId):
7  queries = PROJECT distinct[$2](query);

9  # relevantDocs (QueryId, DocId):
10 relevantDocs = PROJECT[$1,$3](JOIN[$1=$1](queries, relevant));

12 # relColl (Term, DocId):
13 relColl = PROJECT[$3,$4](JOIN[$2=$2](relevantDocs, coll));

15 # distinct collection
16 distinctColl = PROJECT distinct(coll);

18 ################################################
19 # Part 2: Term probabilities and aggregation
20 # Part 2.1: Term probabilities base on document space

22 p_t_c = BAYES df[](PROJECT[$1](coll));
23 p_t_r = BAYES df[](PROJECT[$1](relColl));
```

```
25 # idf for whole collection ( idf_c ) and
26 # idf for the collections constructed from relevant documents ( idf_r ).
27  idf_c  = BAYES max_idf[](PROJECT all[$1](coll));
28  idf_r  = BAYES max_idf[](PROJECT all[$1](relColl));

30 # Query term probabilities :
31 wQuery_c = PROJECT all[$1,$2](SELECT[$1=$3](JOIN(query, idf_c)));
32 wQuery_r = PROJECT all[$1,$2](SELECT[$1=$3](JOIN(query, idf_r)));
33 norm_wQuery_c = BAYES[$2](wQuery_c);
34 norm_wQuery_r = BAYES[$2](wQuery_r);

36 # Part 2.2: Aggregation of query term probabilities
37 wQuery_subsumed    = SUBTRACT subsumed (wQuery_c, wQuery_r);
38 wQuery_independent = SUBTRACT independent(wQuery_c, wQuery_r);
39 norm_wQuery_subsumed = SUBTRACT subsumed(norm_wQuery_c, norm_wQuery_r);
40 norm_wQuery_independent = SUBTRACT independent(norm_wQuery_c, norm_wQuery_r);

42 ################################################
43 # Part 3: Retrieval

45 # Set wQuery and indexColl according to strategy . For example:
46 wQuery = wQuery_subsumed;
47  collIndex  =  distinctColl  ;

49  bir_retrieve   = PROJECT disjoint[$4,$2](JOIN[$1=$1](wQuery, collIndex));
```

```
1  -- Part 1: Basic declarations :
2  CREATE VIEW queries AS SELECT QueryId FROM query;

4  CREATE VIEW relevantDocs AS
5       SELECT QueryId, DocId FROM queries, Relevant
6       WHERE queries.Queryid = Relevant.QueryId;

8  CREATE VIEW relColl AS
9       SELECT Coll.Term, Coll.DocId FROM relevantDocs, coll
10      WHERE relevantDocs.DocId = coll.DocId;

12 CREATE VIEW distinctColl AS SELECT DISTINCT Term, DocId FROM coll;
13 ------------------------------------------------------------
14 -- Part 2: Term probabilities and their aggregation :
15 -- Part 2.1: Term probabilities :
16 CREATE VIEW idf_c AS
17      SELECT Term FROM coll ASSUMPTION MAX_IDF EVIDENCE KEY ();
18 CREATE VIEW idf_r AS
19      SELECT Term FROM relColl ASSUMPTION MAX_IDF EVIDENCE KEY ();

21 CREATE VIEW wQuery_c AS
22      SELECT Term, QueryId FROM query, idf_c WHERE query.Term = idf_c.Term;
23 CREATE VIEW wQuery_r AS
24      SELECT Term, QueryId FROM query, idf_r WHERE query.Term = idf_r.Term;
```

```
26 CREATE VIEW norm_wQuery_c AS
27     SELECT Term, QueryId FROM wQuery_c
28     ASSUMPTION DISJOINT EVIDENCE KEY (QueryId);
29 CREATE VIEW norm_wQuery_r AS
30     SELECT Term, QueryId FROM wQuery_r
31     ASSUMPTION DISJOINT EVIDENCE KEY (QueryId);

33 -- Part 2.2: Term probability aggregation:
34 CREATE VIEW wQuery_subsumed AS
35     wQuery_c MINUS SUBSUMED wQuery_r;
36 CREATE VIEW norm_wQuery_subsumed AS
37     norm_wQuery_c MINUS SUBSUMED norm_wQuery_r;
38 CREATE VIEW wQuery_independent AS
39     wQuery_c MINUS INDEPENDENT wQuery_r;
40 CREATE VIEW norm_wQuery_independent AS
41     norm_wQuery_c MINUS INDEPENDENT norm_wQuery_r;
42 ------------------------------------------------------------
43 -- Part 3: Retrieval:
44 -- Set wQuery and collIndex according to strategy. For example:
45 CREATE VIEW wQuery AS SELECT Term, QueryId FROM wQuery_subsumed;
46 CREATE VIEW collIndex AS SELECT Term, DocId FROM distinctColl;

48 CREATE VIEW birm_retrieve AS
49     SELECT DISJOINT DocId, QueryId FROM wQuery, collIndex
50     WHERE wQuery.Term = collIndex.Term;
```

The programs are structured in three parts: 1. A basic declaration block. 2. The definition of the main probabilistic relations. 3. The definition of relation "*bir_retrieve*" to the retrieval result.

To understand the meaning of the PRA/PSQL programs, consider the document and query representations in Table 4.15. There are ten documents with twenty term-document tuples, one query with 2 query terms and four relevant documents for this query.

Central to the implementation of the BIR model are the two probabilistic relations *idf*_c and *idf*_r: *idf*_c is the discriminativeness of a term in the collection (c denotes the collection), and *idf*_r is the discriminativeness of a term in the set of relevant documents (r denotes the set of relevant documents). The relations *idf*_c and *idf*_r are based on the document-based probabilities $P_D(t|c)$ and $P_D(t|r)$, i.e. the probabilities that term t occurs in the respective set of documents.

The occurrence-based probabilities $P_D(t|c)$ and $P_D(t|r)$ are generated by the new probabilistic relational operator, the relational BAYES. Basically, the relational BAYES performs a computation that leads to the estimate $P_D(t|x) = \frac{n_D(t,x)}{N_D(x)}$, where x is either the collection c or the set r of relevant documents, $n_D(t,x)$ is the number of documents containing term t in set x, and $N_D(x)$ is the total number of documents.

coll

Prob	Term	DocId
1.0	sailing	doc1
1.0	boats	doc1
1.0	sailing	doc2
1.0	boats	doc2
1.0	sailing	doc2
1.0	sailing	doc3
1.0	east	doc3
1.0	coast	doc3
1.0	sailing	doc4
1.0	boats	doc5
1.0	sailing	doc6
1.0	boats	doc6
1.0	east	doc6
1.0	coast	doc6
1.0	sailing	doc6
1.0	boats	doc6
1.0	boats	doc7
1.0	east	doc8
1.0	coast	doc9
1.0	sailing	doc10

relevant

Prob	QueryId	DocId
1.0	q1	doc2
1.0	q1	doc4
1.0	q1	doc6
1.0	q1	doc8

query

Prob	QueryId	DocId
1.0	sailing	q1
1.0	boats	q1

Table 4.15: Representations of collection, query and relevant information

For our running example, we obtain the relations in Table 4.16:

p_t_c	
Prob	Term
0.60000	sailing
0.50000	boats
0.30000	east
0.30000	coast

p_t_r	
Prob	Term
0.75000	sailing
0.50000	boats
0.50000	east
0.25000	coast

*idf*_c	
Prob	Term
0.424283	sailing
0.575717	boats
1.000000	east
1.000000	coast

*idf*_r	
Prob	Term
0.207519	sailing
0.500000	boats
0.500000	east
1.000000	coast

Table 4.16: Probabilities in collection and relevant set

There are $n_D(sailing, c) = 6$ sailing documents, and $N_D(c) = 10$. Then, for example, $P_{\text{p\_t\_c}}(sailing) = 6/10 = 0.6$, $P_{\text{p\_t\_c}}(boats) = 5/10 = 0.5$, and $P_{\text{p\_t\_r}}(sailing) = 3/4 = 0.75$. $P_{\text{p\_t\_r}}(boats) = 2/4 = 0.5$. The expressions with BAYES max_idf perform an idf-based probability estimation. This corresponds to a normalization of the form $\log(P_{\text{p\_t\_c}}(t))/\log(P_{\text{p\_t\_c}}(t_{min}))$ and yields, for example, $P_{\text{idf\_c}}(sailing) \approx 0.42$, and $P_{\text{idf\_c}}(boats) \approx 0.57$.

The query terms are weighted with the IDF-based probabilities. This leads to two relations in table 4.17:

wQuery_c		
Prob	Term	QueryId
0.424283	sailing	q1
0.575717	boats	q1

wQuery_r		
Prob	Term	QueryId
0.207519	sailing	q1
0.500000	boats	q1

Table 4.17: Relation of weighted query

Next, we approach the critical step, namely the aggregation of the query term probabilities $idf(t,c) - idf(t,r)$. If $idf(t,c) \geq idf(t,r)$, then term t is more likely to occur in relevant documents

than in the documents of the collection; otherwise term t tends to occur more in the collection than relevant documents.

The IDF-based probability $idf(t,x)$ can be viewed as $P(t \quad informative|x)$. For the probabilistic subtraction, it can be computed based on two probability assumptions: subsumed and independent. For the subsumed case $F1 = idf(t,c) - idf(t,r)$ when $idf(t,c) > idf(t,r)$, and F1=0 when $idf(t,c) < idf(t,r)$. For the independent case $F1 = idf(t,c) \cdot (1 - idf(t,r))$. This yields the weighted query terms in Table 4.18:

wQuery_subsumed		
Prob	Term	QueryId
0.216765	sailing	q1
0.075717	boats	q1

wQuery_independent		
Prob	Term	QueryId
0.336237	sailing	q1
0.287858	boats	q1

Table 4.18: Query terms' F1 weight in BIR model

For example, the probability for boats in the subsumed case is $0.57 - 0.5 = 0.07$, and in the independent case, we obtain $0.57 * (1 - 0.5) \approx 0.28$. This example illustrates the numerical effect of the probabilistic assumption. This effect is even stronger for normalized query term probabilities, as we illustrate next.

Normalized query term probabilities are based on a disjoint and exhaustive space of events (i.e. sum over probabilities equal to 1.0). This normalization forms an alternative to the non-normalized "*wQuery*" relations, which allows the disjoint projection in the retrieval model.

Note that due to the normalization, boats is viewed as more discriminative (rare) in the relevant documents than in the collection (probability of boats in "*norm_wQuery_r*" greater than in "*norm_wQuery_c*"). Therefore, in the subsumed case, the probability zero is assigned to boats since it is viewed as a poor term to retrieve relevant documents. This makes sense as BIR model prefers the terms that occur more in relevant documents and less in non-relevant documents.

This section illustrates the generation of the core probabilistic relations applied for implementing the BIR model in a probabilistic relational reasoning framework. The relations shown in this section explain the effect of the PSQL/PRA programs. The high-level abstraction of retrieval functions leads to optimal flexibility and re-usability, and these are the main motivations for modelling IR in a probabilistic relational framework.

norm_wQuery_c		
Prob	Term	QueryId
0.424283	sailing	q1
0.575717	boats	q1

norm_wQuery_r		
Prob	Term	QueryId
0.293305	sailing	q1
0.706695	boats	q1

norm_wQuery_subsumed		
Prob	Term	QueryId
0.130978	sailing	q1
0.000000	boats	q1

norm_wQuery_independent		
Prob	Term	QueryId
0.299839	sailing	q1
0.168861	boats	q1

Table 4.19: Normalized query term weight

4.2.4 Language Modelling

In this section we show how to model LM with linear mixture. The term weight is shown as follows:

$$w_{LM} = \log\left(\lambda \cdot P_L(t|d) + (1-\lambda) \cdot P_L(t|c)\right)$$

LM term weight is linear mixture from probability that the term occurs in document and probability that the term occurs in collection. The event spaces of these two probability are tuple (location) spaces, which is indicated by the L subscript.

The PSQL script implemented of LM is as follows:

```
1  -- PSQL: LM retrieval
2  -- Extensional relations :
3  -- coll(Term, DocId); query(Term, QueryId); tf_sum(Term, DocId); mixture(name);

5  -- mixture:
6  DELETE FROM mixture;
7  INSERT INTO mixture VALUES
8  0.8 ('p_t_d'), 0.2 ('p_t_c');

10 CREATE VIEW lambda1 AS
11         SELECT FROM mixture
12         WHERE mixture.name = 'p_t_d';
13 CREATE VIEW lambda2 AS
14         SELECT FROM mixture
15         WHERE mixture.name = 'p_t_c';
```

```
17 -- P(t|d):
18 -- Principle description via views:
19 CREATE VIEW tfCollSpace AS
20         SELECT Term, DocId
21         FROM coll
22         EVIDENCE KEY (DocId);
23 CREATE VIEW p_t_d AS
24         SELECT DISJOINT Term, DocId
25         FROM tfCollSpace;

27 -- For efficiency ,
28 -- bind p_t_d to extensional instance .
29 CREATE VIEW p_t_d AS
30         SELECT Term, DocId
31         FROM tf_sum;

33 -- P(t|c):
34 CREATE VIEW p_t_c_evidence AS
35         SELECT Term
36         FROM coll
37         EVIDENCE KEY ();
38 CREATE VIEW p_t_c AS
39         SELECT DISJOINT Term
40         FROM p_t_c_evidence;

42 -- retrieved(DocId, QueryId):
43 -- Needed for generating schema-compatible views docModel and collModel.
44 CREATE VIEW doc AS
45         SELECT DISTINCT DocId
46         FROM coll;

48 CREATE VIEW docModel AS
49         SELECT Term, DocId
50         FROM lambda1, p_t_d;

52 CREATE VIEW collModel AS
53         SELECT Term, DocId
54         FROM lambda2, p_t_c, doc;

56 -- combine document and collection models
57 CREATE VIEW lm1_p_t__c_d AS
58         docModel UNION DISJOINT collModel;

60 -- retrieve documents
61 CREATE VIEW lm1_retrieve AS
62         SELECT SUM_LOG DocId, QueryId
63         FROM query, lm1_p_t__c_d
64         WHERE query.Term = lm1_p_t__c_d.Term;

66 CREATE VIEW retrieve AS
67         SELECT DocId, QueryId
```

```
68      FROM lm1_retrieve;
```

The PSQL script shows that the probabilities in views "*p_t_d*" and "*p_t_c*" correspond to $P_L(t|d)$ and $P_L(t|c)$ respectively. Similar to the TF-IDF script, we show the principle generation of $P_L(t|d)$, which we then overwrite with a view that takes advantage of a materialized relation "*tf_sum*" that contains the pre-computed probabilities. This is purely for reasons of efficiency, since the view "*tf*" requires an aggregation of probabilities, and this aggregation can be pre-computed in a materialized relation.

Equivalent PRA translation is as the follows:

```
1  # PRA: lm retrieval
2  # Extensional relations :
3  # coll(Term, DocId); query(Term, QueryId); tf_sum(Term, DocId); mixture(name);

5  # Mixture:
6   _delete(mixture);
7  0.8 mixture(p_t_d);
8  0.2 mixture(p_t_c);
9  lambda1 = PROJECT[](SELECT[$1=p_t_d](mixture));
10 lambda2 = PROJECT[](SELECT[$1=p_t_c](mixture));

12 # P(t|d): p_t_d(Term, DocId):
13 tfCollSpace = BAYES[$2](coll);
14 p_t_d = PROJECT disjoint[$1,$2](tfCollSpace);

16 # Optional usage of pre-computed tf:
17 p_t_d = tf_sum;

19 # P(t|c): p_t_c(Term):
20 collSpace = BAYES[](PROJECT[$1](coll));
21 p_t_c = PROJECT disjoint[$1](collSpace);

23 # Retrieved documents for the generation of the collection model that can be
24 # united with the document model.
25 # retrieved(DocId):
26 doc = PROJECT distinct[$2](coll);

28 # Document model:
29 # docModel(Term, DocId):
30 docModel = JOIN[](lambda1, p_t_d);

32 # Collection model:
33 # collModel(Term, DocId):
34 collModel = JOIN[](lambda2, JOIN[](p_t_c, doc));

36 # Combination of docModel and collModel:
37 lm_term_weight = UNITE disjoint(docModel, collModel);
```

```
39 # Retrieve documents:
40 lm_retrieve = PROJECT sum_log[$4,$2](JOIN[$1=$1](query, lm_term_weight));

42 retrieve = lm_retrieve ;
```

The PSQL views correspond to their respective PRA equations. The view "*collModel*" involves an expensive join of query term weights based on $P(t|c)$ with the the whole collection. This join is required since the relational union requires schema-compatible relations "*docModel*" and "*collModel*". The relations "*docModel*" and "*collModel*" are shown in Table 4.20.

The implementation shown above is semantically correct but not efficient, because of the required schema compatibility. We have started to look into an alternative mathematical formulation 4.3, and we have defined an extended PRA with special mixture "*JOIN*" according to the mathematical formulation. With assumption "mixed", the join probability is not the product of probability of P_a and P_b, but $P(\tau) = \frac{P_a(\tau')}{P_a(\tau')+P_b(\tau'')}$. The new mixture "*JOIN*" supports a correct and efficient implementation of LM.

$$RSV_{\text{LM-alt}} = \sum_{t \in d \cap q} \log \frac{\lambda P_L(t|d) + (1-\lambda)P_L(t|c)}{(1-\lambda)P_L(t|c)} \tag{4.2}$$

$$= -\sum_{t \in d \cap q} \log \frac{(1-\lambda)P_L(t|c)}{\lambda P_L(t|d) + (1-\lambda)P_L(t|c)} \tag{4.3}$$

Here we will show the extended LM modelling code:

```
1  --LM with mixed JOIN
2  --P(t|d)
3  CREATE VIEW tfCollSpace AS
4          SELECT Term, DocId
5          FROM coll
6          EVIDENCE KEY (DocId);
7  CREATE VIEW p_t_d AS
8          SELECT DISJOINT Term, DocId
9          FROM tfCollSpace;

11 --P(t|c)
12 CREATE VIEW p_t_c_evidence AS
13         SELECT Term
14         FROM coll
15         EVIDENCE KEY ();
16 CREATE VIEW p_t_c AS
17         SELECT DISJOINT Term
18         FROM p_t_c_evidence;

20 --P(t|d,c)= P(t|c)/(P(t|c)+P(t|d)), assume average document length, and \lamda=0.5
```

p_t_d		
0.500000	sailing	doc1
0.500000	boats	doc1
0.666667	sailing	doc2
0.333333	boats	doc2
0.333333	sailing	doc3
0.333333	east	doc3
0.333333	coast	doc3
1.000000	sailing	doc4
1.000000	boats	doc5
0.333333	sailing	doc6
0.333333	boats	doc6
0.166667	east	doc6
0.166667	coast	doc6
1.000000	boats	doc7
1.000000	east	doc8
1.000000	coast	doc9
1.000000	sailing	doc10

p_t_c	
0.400000	sailing
0.300000	boats
0.150000	east
0.150000	coast

docModel		
0.400000	sailing	doc1
0.400000	boats	doc1
0.533333	sailing	doc2
0.266667	boats	doc2
0.266667	sailing	doc3
0.266667	east	doc3
0.266667	coast	doc3
0.800000	sailing	doc4
0.800000	boats	doc5
0.266667	sailing	doc6
0.266667	boats	doc6
0.133333	east	doc6
0.133333	coast	doc6
0.800000	boats	doc7
0.800000	east	doc8
0.800000	coast	doc9
0.800000	sailing	doc10

collModel		
0.080000	sailing	doc1
0.080000	sailing	doc2
0.080000	sailing	doc3
0.080000	sailing	doc4
0.080000	sailing	doc5
0.080000	sailing	doc6
0.080000	sailing	doc7
0.080000	sailing	doc8
0.080000	sailing	doc9
0.080000	sailing	doc10
0.060000	boats	doc1
0.060000	boats	doc2
0.060000	boats	doc3
0.060000	boats	doc4
0.060000	boats	doc5
0.060000	boats	doc6
0.060000	boats	doc7
0.060000	boats	doc8
0.060000	boats	doc9
0.060000	boats	doc10
⋮		

Table 4.20: Document model and collection model

lm_term_weight		
0.480000	sailing	doc1
0.460000	boats	doc1
0.613333	sailing	doc2
0.326667	boats	doc2
0.346667	sailing	doc3
0.296667	east	doc3
0.296667	coast	doc3
0.880000	sailing	doc4
0.860000	boats	doc5
0.346667	sailing	doc6
0.326667	boats	doc6
0.163333	east	doc6
0.163333	coast	doc6
0.860000	boats	doc7
0.830000	east	doc8
0.830000	coast	doc9
0.880000	sailing	doc10
0.080000	sailing	doc5
0.080000	sailing	doc7
0.080000	sailing	doc8
0.080000	sailing	doc9
0.060000	boats	doc3
0.060000	boats	doc4
0.060000	boats	doc8
0.060000	boats	doc9
0.060000	boats	doc10
0.030000	east	doc1
0.030000	east	doc2
0.030000	east	doc4
0.030000	east	doc5
0.030000	east	doc7
0.030000	east	doc9
0.030000	east	doc10
0.030000	coast	doc1
0.030000	coast	doc2
0.030000	coast	doc4
0.030000	coast	doc5
0.030000	coast	doc7
0.030000	coast	doc8
0.030000	coast	doc10

lm_retrieve		
0.220800	doc1	q1
0.200356	doc2	q1
0.113244	doc6	q1
0.068800	doc7	q1
0.068800	doc5	q1
0.052800	doc4	q1
0.052800	doc10	q1
0.020800	doc3	q1
0.004800	doc9	q1
0.004800	doc8	q1

Table 4.21: Term weights and document RSVs in LM

```
21 CREATE VIEW p_t_dc AS
22   SELECT Term, DocId
23   FROM p_t_c JOIN RATIONAL p_t_d
24   WHERE p_t_c.Term = p_t_d.Term;



27 -- 1-\Prod_t P(t|d,c)
28 CREATE VIEW p_d_dc AS
29  SELECT PROD DocId, QueryId
30  FROM query, p_t_dc
31  WHERE query.Term = p t dc.Term;



34 CREATE VIEW LM AS
35  SELECT COMPLEMENT DocId, QueryId
36  FROM p_d_dc;
```

Corresponding PRA code:

```
1  ###LM with mixed JOIN

3  tfCollSpace  = BAYES[$2](coll);
4  p_t_d  = PROJECT disjoint[$1,$2](tfCollSpace);

6  # P(t|c):  p_t_c (Term):
7  collSpace  = BAYES[](PROJECT[$1](coll));
8  p_t_c  = PROJECT disjoint[$1](collSpace);

10 #p_t_dc
11 p_t_dc  = JOIN RATIONAL [$1=$1] (p_t_c,p_t_d);

13 #lm
14 p_d_dc=PROJECT SUM_LOG [$5,$2](JOIN(query,p_t_dc));
15 lm=PROJECT COMPLEMENT [$1=$2] (p_d_dc);
```

Relations "*p_t_d*" and "*p_t_c*" are identical to the previous implementation, but we can see that there is no need to join the "*p_t_c*" with the whole collection.

4.2.5 BM25

BM25 formula is defined as in equation 4.4:

$$RSV_{BM25}(d,q) = \sum_{t \in q} w_t + k_2 \cdot ql \cdot \frac{avgdl - dl}{avgdl + dl} \tag{4.4}$$

$$w_t = s_1 \cdot s_3 \cdot \frac{tf^c}{K^c + tf^c} \cdot w^{(1)} \cdot \frac{qtf}{k_3 + qtf} \tag{4.5}$$

$$K = k_1 \cdot ((1-b) + b\frac{dl}{avgdl}) \tag{4.6}$$

p_t_dc			
0.444444	sailing	sailing	doc1
0.375000	sailing	sailing	doc2
0.545455	sailing	sailing	doc3
0.285714	sailing	sailing	doc4
0.545455	sailing	sailing	doc6
0.285714	sailing	sailing	doc10
0.375000	boats	boats	doc1
0.473684	boats	boats	doc2
0.230769	boats	boats	doc5
0.473684	boats	boats	doc6
0.230769	boats	boats	doc7
0.310345	east	east	doc3
0.473684	east	east	doc6
0.130435	east	east	doc8
0.310345	coast	coast	doc3
0.473684	coast	coast	doc6
0.130435	coast	coast	doc9

lm		
0.997240	doc3	q1
0.996639	doc6	q1
0.982987	doc9	q1
0.982987	doc8	q1
0.972222	doc1	q1
0.968447	doc2	q1
0.946746	doc7	q1
0.946746	doc5	q1
0.918367	doc4	q1
0.918367	doc10	q1

Table 4.22: Term weights and document RSVs in alternative modelling of LM

The weight $w^{(1)}$ in BM25 is RSJ term weight as introduced in section 2.4.2. Here we use F1 weight to replace it in order to simplify the implementation. The F1 implementation is shown in section 4.2.3. therefore we demonstrate only the remaining part in this section. The whole BM25 is F1 weight multiplied with TF weight, and plus a document length corrector for the whole document.

```
1  #coll (Term, DocId);
2  #para(name); keep the parameters;
3  #wQuery(term); see section 4.2.3

5  0.004 para(k2);
6  1.2 para(k1);
7  0.5 para(b1); #b1+b2=1
8  0.5 para(b2);

10 #average document length of the collection
11 doc = PROJECT distinct [$2] (coll);

13 dl = PROJECT disjoint [$2] (coll);

15 avgdl_coll = PROJECT disjoint [](JOIN [$1=$1](doc_length, BAYES disjoint [] (doc)));

17 #TF normalize, currently parameter c is set to 1
```

```
18 # if we take all the document as same length the K will be 1
19 # 1 K()
20 # we take into account the document length in normalization
21 k = PROJECT [$1](
22    JOIN [](
23     UNITE disjoint (
24      JOIN[](doc, select [$1='b1'](para)),
25      JOIN [](JOIN [](dl,PROJECT inverse [](avgdl_coll)), select [$1='b2'](para))
26       ),
27     select [$1='k1'](para)
28     ))

30 #raw TF
31 tf = PROJECT disjoint [$1,$2] (coll);
32 tfn = PROJECT disjoint [$1,$2] (JOIN mixed [$2=$1](tf, k));
33 # tfn can be pre-computed as well for the efficient purpose

35 #TF * BIRM
36 bm25= PROJECT disjoint [$3] (JOIN[$1=$1](wQuery, tfn));

38 #with document length corrector
39 ql = PROJECT disjoint [] (Query);
40 dl_correct = PROJECT [$3](
41    JOIN [](ql,JOIN (SELECT [$1="k2"](para) ,JOIN mixed [$1=$1] (dl,avgdl_coll))));
42 bm25_dl=SUBTRACT subsume (bm25, dl_correct);
```

In this implementation, parameters s_1 and s_3 are not involved, but it is not difficult to incorporate them into the retrieval function by joining these parameters with bm25 relation. For the document length corrector, we adapt it a little to suit probabilistic implementation, without changing the rankings of original formulation. In the original formula, the document length corrector is $k_2 \cdot ql \cdot \frac{avgdl-dl}{avgdl+dl}$. It can be replaced with $-2 \cdot k_2 \cdot ql \cdot \frac{dl}{avgdl+dl}$ as $k_2 \cdot ql$ is constant in all the documents.

$$dl_{corrector} = k_2 \cdot ql \cdot \frac{avgdl - dl}{avgdl + dl} \tag{4.7}$$

$$= k_2 \cdot ql \cdot (1 - \frac{2dl}{avgdl + dl}) \tag{4.8}$$

$$\sim -2 \cdot k_2 \cdot ql \cdot \frac{dl}{avgdl + dl} \tag{4.9}$$

The implementation of BM25 in PSQL is as follows:

```
1 -- wQuery(Term); see section 4.2.3
2 --coll(Term, DocId);
3 -- para(name), para meters for BM25;

5 --para: (b1+b2=1)
```

```
6  DELETE FROM para;
7  INSERT INTO para VALUES
8  0.004('k2') ,1.2( 'k1') ,0.5( 'b1') ,0.5  para('b2');

10 CREATE VIEW b1 AS
11         SELECT FROM para where para.name='b1';
12 CREATE VIEW b2 AS
13         SELECT FROM para where para.name='b2';
14 CREATE VIEW k1 AS
15         SELECT FROM para where para.name='k1';
16 CREATE VIEW k2 AS
17         SELECT FROM para where para.name='k2';

19 --average document length of the collection
20 CREATE VIEW doc AS
21         SELECT distinct DocId from coll;
22 CREATE VIEW docSpace AS
23         SELECT DocId from doc
24                 EVIDENCE KEY(DocId);

26 CREATE VIEW dl AS
27         SELECT disjoint DocId from coll;



30 CREATE VIEW avgdl_coll As
31         SELECT disjoint from doc_length, docSpace
32           where doc_length.docId=docSpace.docId;



35 --TF normalize, currently parameter c is set to 1
36 --if we take all the document as same length the K will be 1
37 --1 K()
38 --we take into account the document length in normalization
39 CREATE VIEW tf_b1 as
40         SELECT docId from doc, b1;
41 CREATE VIEW inverse_avgdl as
42         SELECT inverse docId from avgdl_coll;
43 CREATE VIEW tf_b2 AS
44         SELECT docId from doc_length, inverse_avgdl,b2;
45 CREATE view ktmp as
46          tf_b1  Union disjoint ,  tf_b2 ;
47 CREATE VIEW k as
48         SELECT DocId from ktmp, k2;

50 --raw TF
51 --tfn can be pre-computed as well for the efficient purpose
52 CREATE VIEW tf as
53         SELECT disjoint term,docId from coll;
54 CREATE VIEW tfn as
55         SELECT mixed term,docId from tf,k
56                 where tf.docId=k.docId;
```

```
58 --TF * BIRM
59 CREATE VIEW bm25 AS
60         SELECT docId FROM wQuery,tfn
61                 Where wQuery.Qterm=tfn.term;

63 --with document length corrector
64 CREATE VIEW ql AS
65         SELECT disjoint FROM Query;
66 CREATE VIEW dl_norm AS
67         SELECT mixed docId from dl,avgdl_coll;
68 CREATE VIEW dl_correct AS
69         SELECT docId from dl_norm, ql, k2;
70 CREATE VIEW bm25 AS
71         SUBTRACT subsume bm25, dl_correct;
```

4.2.6 Divergence From Randomness

DFR model has two parts: informativeness and information gain. There are many probabilistic models for these two parts. However, some of them are difficult to model with relational algebra if there are no building functions for probability estimation based on their different distribution assumptions. Therefore, we choose TF-IDF as informativeness, and Laplace law of succession as information gain part, which is suitable for modelling with relational *BAYES*. As a result, the term t's weight will be expressed by $\frac{1}{tf+1} \cdot (tf_n \cdot \log \frac{N}{n})$. Here tf is row term frequency $N_L(t,d)$, tf_n is normalized term frequency, N is number of documents in the collection, and n is number of documents contain term t. If more distribution assumptions, such as Bernoulli or Bose-Einstein model, are to be modeled, then the probabilities need to be calculated outside the PRA/PSQL, and instantiated in advance.

```
1  #coll (Term, DocId); query(Qterm,QueryId);
2  #tf_sum(Term, DocID); document length normalized TF
3  # tf_poissona (Term, DocID); 2-Poisson approximated TF
4  #log N/n is similar to log N+1/n+0.5

6  #INF1 tf*idf
7  idf = BAYES max_idf [](PROJECT [$1](coll));
8  #tf can be any normalized tf
9  tf = tf_sum;
10 Inf1=PROJECT[$1,$5,$2](
11       JOIN[$1=$1](JOIN[$1=$1](query,idf),tf));

13 #INF2 1-tf/( tf +1)
14 Inf2=PROJECT complement [$1,$4,$2](
15               JOIN[$1=$1](query,tf_poissona));
```

```
17 #wQuery=PROJECT [$1,$2,$3](
18         SELECT[$1=$4,$2=$5,$3=$6] (JOIN (Inf1,Inf2)));
19 wQuery=PROJECT [$1,$2,$3](
20         JOIN[$1=$1,$2=$2,$3=$3](Inf1,Inf2));

22 dfr=PROJECT disjoint [$2,$3](wQuery);
```

The corresponding PSQL implementation is :

```
 1 # coll (Term, DocId); query(Qterm,QueryId);
 2 #tf_sum(Term, DocID); document length normalized TF
 3 # tf_poissona (Term, DocID); 2-Poisson approximated TF

 5 --log N/n is similar to log N+1/n+0.5

 7 -- INF1
 8 CREATE VIEW idf AS
 9      SELECT Term FROM coll ASSUMPTION MAX_IDF EVIDENCE KEY ();

11 --tf can be any normalized tf
12 CREATE VIEW tf AS
13          SELECT term, docId from tf_sum;

15 CREATE VIEW inf1 AS
16          SELECT Qterm, docId, QueryId from query,idf,tf
17          Where query.Qterm=;

19 --INF2
20 CREATE VIEW inf2 AS
21          SELECT complement Qterm docId, QueryId from Query,tf_poissona
22          WHERE Query.Qterm=tf_poissona.term and query.Qterm=tf.Term;

24 CREATE VIEW wQuery AS
25          SELECT Qterm, docId, QueryId from inf1, inf2
26          where inf1.Qterm=inf2.Qterm and inf1.docId=inf2.docId and inf1.QueryId=inf2.
                QueryId;

28 CREATE VIEW dfr AS
29          SELECT disjoint docId, QueryId from wQuery;
```

4.3 Modelling Precision and Recall

Precision and recall are frequently used retrieval quality measurements. They can be interpreted as the conditional probabilities $P(\text{relevant}|\text{retrieved})$ and $P(\text{retrieved}|\text{relevant})$ respectively. This interpretation implies that we can model precision and recall in a probabilistic relational framework which supports the description of conditional probabilities. This has two benefits: Firstly,

retrieved	
QueryId	DocId
q1	doc2
q1	doc4
q1	doc6
q1	doc8
q1	doc1
q1	doc3
q1	doc5
q1	doc7
q1	doc9
q2	doc5
q2	doc4

relevant	
QueryId	DocId
q1	doc1
q1	doc4
q1	doc9
q1	doc11
q1	doc14
q1	doc19
q2	doc4

Table 4.23: Retrieved and relevant documents

the measures become part of the conceptual framework in which we model IR. Secondly, by replacing black-box tools that produce precision/recall values, we enable the application-specific modification of measures.

For illustration, consider the following data in relations: "*retrieved*" which is ranking list from search engine and "*relevant*" which is the ground truth for queries:

Based on these extensional relations, we define three views that will be used later for defining precision and recall.

```
1  -- PSQL
2  -- Extensional relations :
3  -- retrieved(QueryId, DocId);
4  -- relevant(QueryId, DocId);

6  CREATE VIEW retrievedSpace AS
7      SELECT QueryId, DocId
8      FROM retrieved
9      ASSUMPTION DISJOINT
10     EVIDENCE KEY (QueryId);

12 CREATE VIEW relevantSpace AS
13     SELECT QueryId, DocId
14     FROM relevant
15     ASSUMPTION DISJOINT
16     EVIDENCE KEY (QueryId);

18 CREATE VIEW retrieved_and_relevant AS
19     SELECT QueryId, DocId
```

```
20      FROM relevant, retrieved
21      WHERE relevant.QueryId = retrieved.QueryId
22      AND relevant.DocId = retrieved.DocId;
```

The view "*retrievedSpace*" contains for each query the probabilistic tuples that reflect the probability that a document is among the retrieved documents of the query. The view "*relevantSpace*" has an analogous role for the relevant documents. Given these spaces and the view "*retrieved_and_relevant*", we describe precision and recall:

```
1   -- PSQL: precision and recall

3   CREATE VIEW precision AS
4     SELECT DISJOINT query
5     FROM retrieved_and_relevant, retrievedSpace
6     WHERE retrieved_and_relevant.QueryId =
7           retrievedSpace.QueryId
8     AND retrieved_and_relevant.DocId =
9           retrievedSpace.DocId;

11  CREATE VIEW recall AS
12    SELECT DISJOINT query
13    FROM retrieved_and_relevant, relevantSpace
14    WHERE retrieved_and_relevant.QueryId =
15          relevantSpace.QueryId
16    AND retrieved_and_relevant.DocId =
17          relevantSpace.DocId;
```

The translation of the first PSQL script with views "*retrievedSpace*" and "*relevantSpace*" yields the following PRA program:

```
1   # PRA
2   retrievedSpace = BAYES[$1](retrieved);
3   relevantSpace = BAYES[$1](relevant);

5   retrieved_and_relevant = PROJECT[$1,$2](JOIN[$1=$1,$2=$2](relevant, retrieved));
```

The first two equations yield the two spaces "*retrievedSpace*" and "*relevantSpace*", where in each space a document occurs with the probability $P_{space}(d|q) = 1/N(q)$, where $N(q)$ is the number of documents for query q. The third equation yields the relation of retrieved and relevant documents.

The third equation yields the relation of retrieved and relevant documents in Table 4.24.

Next, consider the PRA equations for precision and recall:

```
1   # PRA: precision and recall
```

retrievedSpace		
Prob	QueryId	DocId
1/9	q1	doc2
1/9	q1	doc4
1/9	q1	doc6
1/9	q1	doc8
1/9	q1	doc1
1/9	q1	doc3
1/9	q1	doc5
1/9	q1	doc7
1/9	q1	doc9
1/2	q2	doc5
1/2	q2	doc4

relevantSpace		
Prob	QueryId	DocId
1/6	q1	doc1
1/6	q1	doc4
1/6	q1	doc9
1/6	q1	doc11
1/6	q1	doc14
1/6	q1	doc19
1	q2	doc4

retrieved_and_relevant		
Prob	QueryId	DocId
1	q1	doc1
1	q1	doc4
1	q1	doc9
1	q2	doc4

Table 4.24: Retrieved and relevant spaces

```
3 precision = PROJECT disjoint[$1](JOIN[$1=$1,$2=$2](retrieved_and_relevant,
      retrievedSpace ));
4 recall = PROJECT disjoint[$1]( JOIN[$1=$1,$2=$2](retrieved_and_relevant,
      relevantSpace ));
```

The joins of "retrieved_and_relevant" with the respective spaces, followed by disjoint projections, yield the precision and recall values in Table 4.25:

precision	
Prob	QueryId
3/9	q1
1/2	q2

recall	
Prob	QueryId
3/6	q1
1	q2

Table 4.25: Precision and Recall Relations

4.4 Summary

In this chapter, we have demonstrated the implementation of five main models, namely TF-IDF, BIRM, LM, BM25, and DFR. In addition, the modelling of precision and recall has been discussed.

For TF-IDF and LM, we showed semantically correct implementations, whereas the BIRM

implementation does not implement the genuine BIRM formulation. For DFR, we only model the informativeness based on the assumption that term is independent to all other tokens.

The implementation of the retrieval models and their variations demonstrates that our probabilistic modelling is flexible regarding probability estimation and aggregation; it is applicable to large-scale data; and it allows to formulate and investigate retrieval models in an abstract, relatively compact but still efficient representation.

The implementation of evaluation measures (precision and recall) shows that the quality measures can also be embedded into the conceptual framework of probabilistic relational modelling. The expressiveness of relational modelling allows us to customize the measures and perform post-processing of the retrieval result.

Chapter 5

Context-specific Frequencies in Probabilistic Retrieval Models

---

Term weighting in probabilistic models is based on the term occurrence frequency within a certain context. The "context" has two meanings: 1. the part of a text or statement that surrounds a particular word or passage and determines its meaning; 2. the circumstances in which an event occurs, including user information need, searching environment etc.

The second definition is more general and more difficult to be incorporated into retrieval models. For example, when a user types in a query "*quantum computing*", he prefers to know algorithms in computer graphics rather than an explanation of the quantum computing in modern physics. The retrieval system would not be able to know this, if there is no user profile, search history, or explicit user specification to show user searching preference. Therefore, the "*context*" in this thesis refers the text surrounding a certain word, it is a physical association rather than a semantic one. This is different to the concept of context in [Azzopardi, 2005] which explicitly assumes that the context is a set of semantically associated documents.

The "*context*" here is also remarkably different to the fixed text block in passage retrieval[Kaszkiel and Zobel, 1997], which divides a document into smaller chunks (i.e.passage), weights the terms according to the chuck where the terms appear, then combines the evidence from the passage to score the document. In our study, the context scope is decided by retrieval objects, it can be a document, a set of documents, or a few sets of document. When the context involves a few sets of documents, the documents in a set do not concern the same subject,

although there is such a possibility when the collections are organized by subject. Later, we will also study whether the set documents have same subject will impact the retrieval quality of context-frequencies based retrieval model.

In this chapter we investigate the retrieval qualities of the models utilizing the probabilities estimated from frequencies within specific surrounding texts, i.e., context-specific frequencies, which also denotes the frequencies counted base on different elements rather than traditional document or token based frequencies. The whole study of this chapter is based on the structured document collection INEX[Fuhr et al., 2003b] to benefit the structure information. However, we do not consider the structure characteristic of the element of a structured document, i.e. title, body, as [Robertson et al., 2004] do.

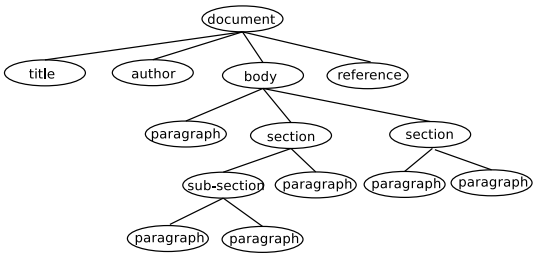
This chapter is structured as follows. In section 5.1 we introduce the basic concepts in structured document retrieval, which is closely related to the motivation of context-specific frequencies. In section 5.2 we brief the two main motivations for us to investigate the context-specific frequencies. In section 5.3 we discuss the application of the context-specific frequencies in document and collection ranking in multi-collections. In section 5.4, related works on structured document retrieval are introduced. In section 5.5 and section 5.6, we define the context-specific frequencies, and the retrieval state value for the document and element based on context-specific frequencies. In section 5.7, we run the experiment on a structured document collection with the retrieval function defined in section 5.6. As we postulate that whether a set documents concerning same subject would affect the retrieval result, we conduct another experiment which hires the same methodology, while the original document collection has been mixed and divided randomly into sub-collections. Detailed results and analysis are also described in this section. In section 5.8 we investigate an alternative distinctiveness measurement, and compare it with traditional IDF. At the end of this chapter we summarize the main findings in section 5.9.

5.1 Structured Document Retrieval

The idea of context-specific frequencies was initially motivated by structured document retrieval, therefore, we would like first to introduce structured document retrieval before we start context-specific frequencies.

Structured documents, XML documents, are widely used for information representation and exchange. They contain tags which explicitly divide the documents into logical parts: title,

author, abstract, body, section, subsection, paragraph, and list etc. These logical parts are called the elements of the structured documents, which are different from the segmentation defined by physical size. Some elements must be nested in specific elements, for example, document contains title, author, abstract and body, body contains sections and paragraphs, section contains subsections, paragraphs, and so on. This constitutes a tree-structure, in which the leaf node of a document tree is a paragraph , a list, or a table etc, the inner node of the tree (or intermediate element) is a section, a subsection etc, and the root node of the tree is a document. In some XML documents, the tags have semantic meaning, which can be helpful for retrieval. However, such tags are designed for special information needs, and can not be applied generally. Therefore, our study focuses on the structural tags. Figure 5.1 shows the structure and code of an XML document example.



(a) Structure of XML

```
<article>
    <title>
    TF-IDF Uncovered
    </title>
    <author>
    Thomas Roelleke , Jun Wang
    </author>
    <abstract>
    Interpretations of TF-IDF are based on
    </abstract>
    <section>
        <paragraph>
        Introduction and background...
        </paragrah>
        <subsection>
        IR fundations...
        </subsection>
        <subsection>
        Models....
        </subsection>
    </section>
    <section>
    Summary...
    </section>
</article>
```

(b) Example of XML

Figure 5.1: An example of a XML document

The structured information of XML documents makes it possible to retrieve smaller objects than a whole document, such as sections, paragraphs etc, which is termed as element retrieval. There are two main types of element retrieval. For the first one, the users specify only the subject

that they want to know, for example "relevance feedback", they do not care in which part of the document the words occur. The retrieval system will try to find the most relevant elements and return them to the user. The second one requires users to specify both content and structure need, for example the users want information about "relevance feedback", also they want it to occur in the title or the abstract of a document. For this kind of query, the system will return to the users the specific elements containing "relevance feedback". However, the important issue for both types of the element retrieval is how to rank the element. Therefore our later work concentrates on content only retrieval.

In the next section we discuss the reason that we look into context-specific frequencies.

5.2 Motivation of Context-specific Frequencies in Structure Document Retrieval

Figure 5.2 illustrates a mini structured document collection $collection_1$ with three documents, each of which has sections, subsections, paragraphs. To simplify the problem, the terms are assigned to the leaves of the documents. For example, term t_1 is assigned to some document leaves of doc_1. A document leaf could be, for example, a paragraph or a table. The inner node of a document tree could be a section or a subsection. The assignment of terms to the inner nodes is usually based on the following aggregation rule[Fuhr et al., 2003a]:

> If term t is assigned to node d', and node d' is a child (direct successor) of node d, in other words, d is the parent (direct ancestor) of d', then t is assigned to d.

We view the issue of whether terms can be directly assigned to inner nodes as being of minor importance, as we can always consider an extra leaf for a node, and this leaf contains the content (term) of the inner node.

Based on the structured document collections, there are two motivations for context-specific frequencies as shown in section 5.2.1 and section 5.2.2

5.2.1 Inverse Document or Element Frequency?

In non-structured document collections, where the documents are streams of terms, the frequencies related to terms are location based (e.g. TF) or document based (e.g. DF). Whereas for the structure document, whose structure information is explicitly represented, the frequencies can be counted based on different element types, e.g. document, section, paragraph, which is also

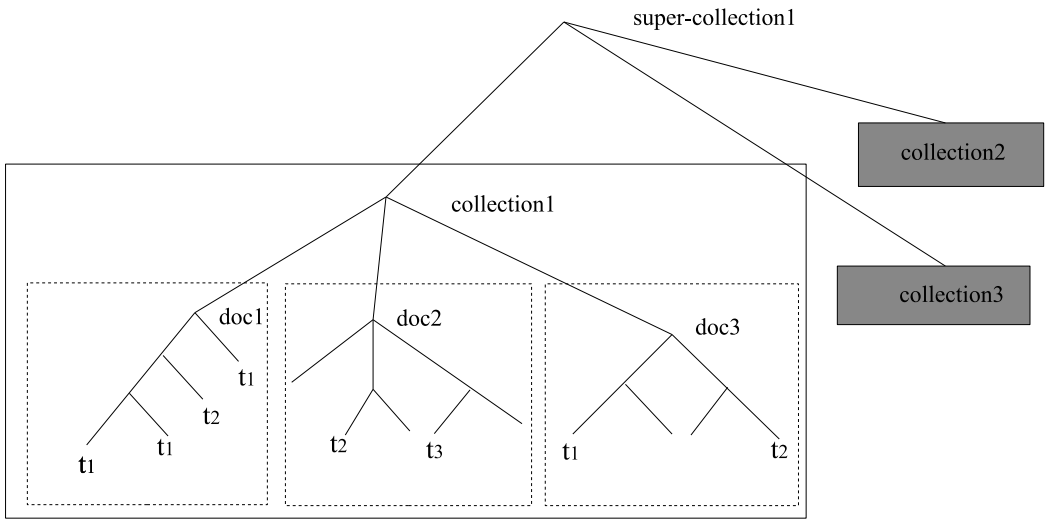


Figure 5.2: An illustration of a mini structured document collection

called element frequency (EF) in general. Considering the documents and terms in figure 5.2, we observe the corresponding document and leaf element frequencies which are shown in table 5.1:

Term	Document frequency (df)	Leaf element frequency (ef)
t_1	2	4
t_2	3	3

Table 5.1: Document frequency and element frequency for the terms in figure 5.2

There is a contradiction when we try to use different inverse frequencies to judge the importance of a term:

> From an IDF-point of view, term t_1 is more discriminative than term t_2: $idf(t_1) > idf(t_2)$.
>
> From an IEF-point of view, term t_2 is more discriminative than term t_1: $ief(t_1) < ief(t_2)$.

This contradicting evidence for the discriminativeness of terms, i.e. the ability of a term that can distinguish relevant and non-relevant document, is the main motivation for carrying out the research on context-specific frequencies. The question is which discriminativeness we should apply for structured document retrieval, IDF or IEF?

5.2.2 Element Ranking with Context-specific Frequencies

Another motivation for context-specific frequencies is that they can effectively rank elements, based on the fact that users do not expect retrieval system return them a long list[Betsi et al., 2006]. Therefore, our strategy for returning object is:

> If an element e and its children e_i are all concerning the same subject s, or say, they all contain the same keywords, then we prefer to rank element e higher than its individual child e_i, which means element e will be assigned higher RSV. If only one of the children (e_n) is about the subject s, then we would like to see only the child e_n rather than the parent e.

If we use IDF which is computed based on the whole collection for ranking an element and its children, then the query term will contribute the same to the RSV of all the elements. It will not be able to differentiate which element is more relevant. However, the IDF computed based on element frequency within their parents can help to solve such a problem. As $ief(t,e)$ for the term t in context of element e will be low if the term t occurs frequently in the children of element e, which leads that the RSV for the e's child will be lower than the RSV of the element e. Accordingly, the element e will gain a higher rank. Similarly, $ief(t,e)$ for the term t in context of element e will be high if the term t occurs in few children of element e, which implies that the RSV's for the e's children will be higher than the RSV of the element e. As a result, the element will gain a lower rank than its children. Without using any parameters, context-specific IDF based RSV can meet the user's ranking requirement.

Given the collection example in figure 5.2 and a query about t_1, we can see that $ief(t_1, doc_1) = -\log\frac{3}{4}$ is smaller than $ief(t_1, doc_3) = -\log\frac{1}{4}$, because t_1 occurs in most of the elements in doc_1. $ief(t_1, doc_1) = -\log\frac{3}{4}$ is also smaller than $idf(t_1, collection_1) = -\log\frac{2}{3}$. This helps to rank document doc_1 and the elements in doc_3 in higher orders. It makes sense as the whole document doc_1 is about t_1, hence, it is better to return the whole document to the user rather than the individual elements. Also as only one element in document doc_3 is about t_1, so it is better to return the single element in a higher rank.

We have discussed the benefit of applying context-specific frequencies to retrieval in a single collection. In the next section we will show the merit of context-specific frequencies in multi-collection retrieval, in term of both document or collection ranking.

5.3 Retrieval in Structured Document Collections

We have introduced the idea that a document collection can be represented as a virtual document tree in figure 5.2. If more such virtual document trees are connected from root to a new virtual root node, then a bigger virtual document tree can be constructed. Figure 5.3 demonstrates such a super tree structure for a multi-collection.

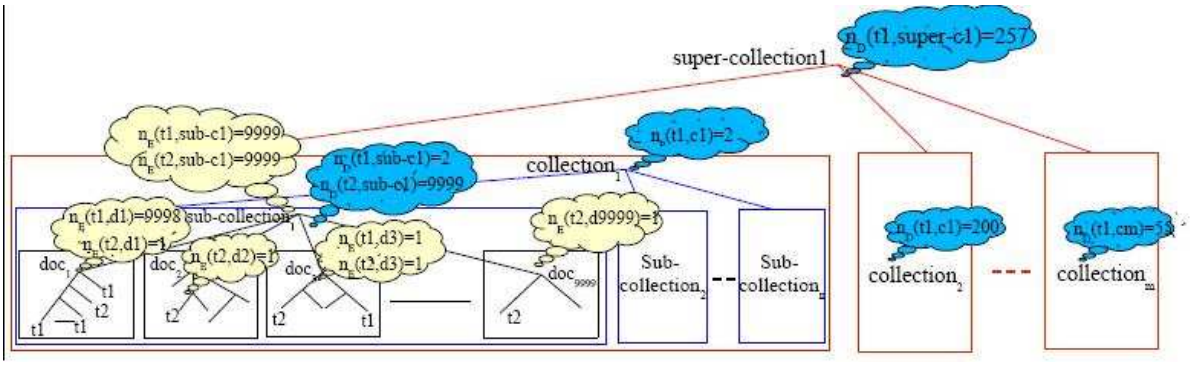


Figure 5.3: Tree-structured document collection

With the assumption that each leaf node is a document, the widely used test collection CACM can be viewed as a small structured document with only 1 level; the INEX IEEE collection can be viewed as a medium size structured document with 2 levels which are journal and year in journal respectively; and the TREC text collection can be viewed a large sized structured document with 3 levels: volume, journal/press/registration in each volume respectively, and year in each journal/press/registration. All these test collections can be formed into a bigger document tree when necessary.

From practical point of view, this virtual document tree provides the retrieval system with great flexibility for context management. All the documents can be viewed as a huge tree-structured document T, where the sub-tree can be a document or a set of documents (a sub-collection), likewise the elements of a document can also be viewed as a sub-tree of T. When a sub-tree or more sub-trees are removed from the virtual document tree T, T is still a document tree. Similarly, one or more sub-trees can be appended to T without affecting the existence of other sub-trees. This tree structure enables the retrieval system to dynamically index or remove data from its data presentation, as well as to retrieve information in the sub-tree starting from any sub-tree's root node. Such mechanics will be of great benefit to distributed retrieval, efficiency issues or security restrictions.

In distributed retrieval, some data collections may be unavailable due to network problems, then the retrieval process can not be run over the whole data collections. Or in order to decrease the overhead of data transmission over the network, only part of the data collections are chosen for retrieval. Even for the centralized the data collection, it can benefit the system efficiency to run the retrieval process on part of the data collection.

For security reasons, access to some data set could be restricted to certain people. According to the user's profile, these data can be removed form the virtual document during the retrieval process. This will not only ensure to exclude certain user from the restricted data, but also avoid inappropriate ranking because of the occurrence frequencies in the restricted data.

For those retrieval systems that wish to continue the service during system updating time, context-specific frequencies provide a system on-line data update ability. During the data updating period, part of the data is quarantined for update, and the retrieval process can be carried out on the rest of the data as usual.

All the cases mentioned above can be viewed as dynamically managing a document collection during retrieval time, and the informativeness of a term is decided by its occurrence probability in a particular document tree, context-specific frequencies. retrieving in the specific context.

The tree structure of the collections defines the element, document and collections with a generalized concept: sub-tree. It also provides a uniform way to rank elements, documents and collections. Documents and elements are both retrievable objects, and a document is the biggest element that can be retrieved. Collections can be a retrieval object too, which can be treated as source selection in distributed retrieval. However, the retrieval of collection is the intermediate step of retrieval, document or element retrieval follows after the promising collections are selected.

In the next two sections, we introduce document and collection ranking in the multi document collections.

5.3.1 Document Ranking with Context-specific IDF in Multi-collection Retrieval

In distributed retrieval, it is widely accepted that the document from the collection where more query terms occur will be more likely to be relevant, and should be assigned high relevant score.

We believe however that the document from the collection containing less query terms should be ranked higher. One of the reasons is the same as what we mentioned in section 5.2.2 about

element retrieval, which prefers to rank higher the element from the document with few query terms. Another reason is based on the decision theory: if a collection has very few documents containing the query terms, and these document are highly relevant to the query, then these documents should be ranked higher than those documents from the collections with more query terms. Because this mechanics makes it possible for users to find relevant documents with less reading.

Following Steven Robertson's PRP, we can prove that ranking the document with fewer query terms higher will be the optimum from the cost point of view.

Assume a_1 and a_2 are the cost for reading a relevant and a non-relevant document respectively, and reading a relevant document costs less than a non-relevant document($a_1 < a_2$). Cost(d,c) is the cost for reading a document d in collection c.

The cost of a user reading a document is:

$$Cost(d,c) = a_1 \cdot P(r|q,d,c) + a_2 \cdot P(\overline{r}|q,d,c) = a_2 + (a_1 - a_2) \cdot P(r|q,d,c) \tag{5.1}$$

As $a_1 - a_2 < 0$, so the more relevant a document is to a query, the less cost to read that document. If the probability of a document being relevant to a query $P(r|q,d,c)$ is estimated by BIR model without relevance information, and purely based on sub-collection frequency, then it will be:

$$P(r|q,d,c) \sim \sum_{t \in q} \log \frac{N_D(c) - n_D(t,c)}{n_D(t,c)} \tag{5.2}$$

Assume collection c_i has more documents containing more query terms than collection c_j, i.e. $n_D(t,c_i) > n_D(t,c_j)$, then $P(r|q,d_n,c_i) < P(r|q,d_m,c_j)$, consequently $Cost(d_n,c_i) > Cost(d_m,c_j)$. This shows that the cost of reading a document from the collection containing less query terms is low.

Similarly to element retrieval, context-specific IDF can automatically rank high the documents from collections containing fewer query terms. Because context-specific IDF for the term from the collection with fewer query terms is high, which leads to high RSV for those documents in the collection with fewer query terms. And context-specific IDF assigns a low RSV to the document in the collection with more query terms.

5.3.2 Collection Ranking with Context-specific Frequencies in Multi-collection Retrieval

In distributed information retrieval, one of the important issues is resource selection (i.e. database selection, or collection selection), which is to choose some collection likely to have more relevant documents. The motivation behind this topic is to save traffic over the network, and decrease the overhead of retrieval in every data collection.

In previous distributed retrieval studies, there are many resource selection algorithms, three representatives are [Yuwono and Lee, 1997], [Gravano and Garcia-Molina, 1995] and [Callan et al., 1995b]. [Callan et al., 1995b] used DF and ICF to choose the collection following the argument of TF-IDF used in document retrieval. [Hawking and Thistlewaite, 1999] confirmed the effectiveness of this method. While [Gravano and Garcia-Molina, 1995] ranked the document collection according to the summary of similarity of each document in that collection to the query ($\sum_{d \in c} sim(d,q)$), and then ranked the documents from top collections with their local similarity scores. [Yuwono and Lee, 1997] mainly used the skewness of distribution of a term to measure its discriminativeness in terms of collection. [Si et al., 2002] applied language modelling in the source selection, the importance of a collection is decided by the probability that the collection generates the query, which is smoothed by the probability that the term occurs in all the collections. All these methods need to maintain global term statistics in the broker of the distributed retrieval system, or probe the term distribution in the sub-collections, or estimate the collection relevant score based on previous query.

However, it is not a simple task to maintain and update the statistical information for all the data collections or relevance history. [Callan et al., 2001] send queries to database and examine the return documents to get the term distributions. Their work enlightens us with a strategy for collection selection. We send a query to each collection, then choose some promising collections according to the returned statistical result about query terms.

In section 5.3.1 we have showed that documents from the collection which contains fewer query terms should be ranked higher from the cost point of view, and IDF can better meet such a requirement. Therefore we use the summary of IDF over the query terms to measure the informativeness of a query in terms of collection ($QInf(q,c)$). If a query is highly informative in a collection, then this collection should be ranked as promising collection.

$$QInf(q,c) = \sum_{t \in q} idf(t,c) \tag{5.3}$$

On the other hand, previous resource selection works suggest that in distributed information retrieval the data collections or resources which have more documents containing the query terms (i.e. high $df(t,c)$) should be chosen [Callan et al., 1995b]. To compare performance of our query informativeness strategy with traditional most occurrence strategy, we also run another collection selection strategy, query document frequency $QDF(q,c)$, which uses the summary of context-specific DF (or local DF) over the query terms to rank the collection .

$$QDF(q,c) = \sum_{t \in q} df(t,c) \tag{5.4}$$

So far, we have argued that context-specific frequencies can rank element, document and collection effectively. Before we go to the details of context-specific frequencies, lets look at related work in structured document retrieval.

5.4 Related Work on Structured Document Retrieval

Structured document retrieval enables the retrieval of elements, whose main issues is to assign a retrieval status value (RSV) to each element. The hierarchy of a structured document makes it more complicated to assign RSVs to the elements. As we illustrated before a document may contain sections and paragraphs, sections may contain sub-sections and paragraphs, and so on.

Here are normally used term weight strategies: either aggregate term weights or aggregate RSV's, or provide an alternative inverse frequency to the classical inverse document frequency.

[Fuhr and Großjohann, 2001] aggregated whole term weight based independent assumption. Whilst [Roelleke et al., 2002] aggregated part of the term weight TF by assigning each child element an access probability, then children's TF weights aggregate to their parent proportionally. [Ogilvie and Callan, 2003], [Ogilvie and Callan, 2004] and [Ogilvie and Callan, 2005] applied language Modelling in term weight aggregation, which linearly combines the probability the term occurs in the element and the probabilities the term occurs all its children or parents.

In RSV aggregation aspect, [Callan, 1994] and [Callan et al., 1995a] gave each passage a TF-IDF based score which indicate its contribution to the document. Their passage score function can be applied to distributed retrieval for source selection with little modification. [Grabs and Schek, 2002] used the distance between two nodes in a document tree to decide the element's weight contribution to its ancestor's. The greater the distance from an element to its ancestor, the less contribution of the elements to their ancestor.

To aggregate the term weight or RSV, each element usually needs to be assigned an aggregation weight, which is not an easy task. Because the aggregation weight is normally estimated from a particular data set, and need to be trained to achieve good retrieval performance. The drawback is that the aggregation weights estimated based on the training data set will not guarantee good performance in any other data sets.

Apart from aggregation method, alternative IDF is another way for element ranking. [Grabs and Schek, 2002] indexed the TF for the leaf nodes (elements) of document tree, and EF for each sub-tree (categorized by subject). According to the query, the system dynamically decides the computation of RSV. They have three models: single collection, multi-collection or nested model. In the first two models, the IDF value is computed based on one or more sub-collections during retrieval time. This retrieval function has great flexibility to respond users' information need. [Mass and Mandelbrod, 2003] indexed TF, IDF for different type elements (document, section, paragraph etc), and computed RSV for each type element based on this type element's inverse element frequency, then merge the results. Although Mass and Mandelbrod had type specific IDF's, these IDF's are computed based on the whole collection, which do not utilize all merits of context-specific frequencies.

Our work is to investigate the retrieval model based on type-specific and context-specific frequencies. It will be generally called context-specific in the later part, as the frequency type and its context are chosen according to the retrieval object.

In the next section 5.5 we will give the definition of the frequencies. And in section 5.6 we will define the retrieval function with context-specific frequencies.

5.5 Context-specific Frequencies Definition

Before we extend the discussion of application of context-specific frequencies, we give a dual and consistent notation and the definition of the frequencies: collection frequency (CF), document frequency (DF), element frequency (EF), location frequency (LF), and the respective inverse frequencies.

Definition 7 *Frequencies:*

t	*a term*
c	*a collection*
$n_C(t,c)$	*number of* sub-collections *in which t occurs*
$N_C(c)$	*number of* sub-collections *in c*
$n_D(t,c)$	*number of* documents *in which t occurs*
$N_D(c)$	*number of* documents *in c*
$n_E(t,c)$	*number of* elements *in which t occurs*
$N_E(c)$	*number of* elements *in c*
$n_L(t,c)$	*number of* locations *in which t occurs*
$N_L(c)$	*number of* locations *in c*

$$cf(t,c) := \frac{n_C(t,c)}{N_C(c)} \tag{5.5}$$

$$df(t,c) := \frac{n_D(t,c)}{N_D(c)} \tag{5.6}$$

$$ef(t,c) := \frac{n_E(t,c)}{N_E(c)} \tag{5.7}$$

$$lf(t,c) := \frac{n_L(t,c)}{N_L(c)} \tag{5.8}$$

$$icf(t,c) := -\log cf(t,c) \tag{5.9}$$

$$idf(t,c) := -\log df(t,c) \tag{5.10}$$

$$ief(t,c) := -\log ef(t,c) \tag{5.11}$$

$$ilf(t,c) := -\log lf(t,c) \tag{5.12}$$

Here, the element E is a generalized representation of elements, which can be any type of element: section, subsection, paragraph... Also the collection frequency that we use is different to some other works which take the total number of a term appearing in a collection as collection frequency, it the number of collections in which a term appears.

The inverse document frequency (IDF) is the highest abstraction in the sense that the possibly multiple occurrence of a term in a document is completely discarded. The inverse element frequency (IEF) discards only the multiple occurrence of a term in an element. The inverse location frequency (ILF) preserves the multiple occurrence of terms. It is also called inverse token frequency (ITF) or inverse collection token frequency (ICTF) in some others' work, we use the name ILF to avoid the confusion with TF. The inverse collection frequency (ICF) measures the importance of term in term of collection selection.

With tree-structure based collection, we can generalize definition 7 for any root of a sub-tree:

Definition 8 *Tree-based frequencies: Let $c_1, \ldots, c_n$ be the children of node c. Then:*

$$n_C(t,c) := \sum_i n_C(t,c_i), \quad N_C(c) := \sum_i N_C(c_i)$$
$$n_D(t,c) := \sum_i n_D(t,c_i), \quad N_D(c) := \sum_i N_D(c_i)$$
$$n_E(t,c) := \sum_i n_E(t,c_i), \quad N_E(c) := \sum_i N_E(c_i)$$
$$n_L(t,c) := \sum_i n_L(t,c_i), \quad N_L(c) := \sum_i N_L(c_i)$$

Definition of cf(t,c), df(t,c), ef(t,c), lf(t,c) and inverse frequencies is strictly analogous to definition 7, 5.6 to 5.12.

Since we consider elements in documents, and we consider locations in elements, we obtain:

$$\forall t : n_C(t,c) \leq n_D(t,c) \leq n_E(t,c) \leq n_L(t,c) \text{ and } N_C(c) \leq N_D(c) \leq N_E(c) \leq N_L(c)$$

For illustrating the difference between IDF and ILF, consider the following example: In a collection with $N_D(c) = 1{,}000$ documents and $N_L(c) = 200{,}000$ locations, the average document length is 200 locations.

Let term t_1 occur in $n_L(t_1,c) = 2{,}000$ locations. Then, in average, t_1 occurs in $lf(t_1,c) = 2{,}000/200{,}000 = 1/100$ locations. This means that if t_1 is evenly distributed, then t_1 occurs in average $N_L(d) \cdot lf(t_1,c)$ times in a document, with $df = 1000/1000 = 1$. And if t_1 is clingy together, then it can occur in only 10 documents with $df = 10/1000 = 1/100$.

As discussed above, IDF is superior to ILF since the burstiness (clinginess) of good terms is reflected by IDF but not by ILF. From this point of view, IEF is expected to be better than ILF, since the same argument holds for IEF versus ILF.

More complex, but actually leading to the motivation for a context-specific discriminativeness, is the discussion of IDF vs IEF. Since IDF considers the burstiness in documents, IDF is more suitable in retrieving documents (large elements, roots of sub-trees), whereas IEF suits smaller granular element retrieval.

In the section 5.6, we generalize the ranking definitions based on a tree-structured collection.

5.6 RSV with Context-specific Frequencies

Section 5.5 shows different type frequencies and discusses their properties. It also reminds us of the work on IDF and inverse token (term) frequency (ITF, same to our ILF) (see

[Church and Gale, 1995] and related publications). They proved that IDF works better than ILF in document retrieval. The main explanation is: Good query terms tend to be not frequently in the collection, but if they occur in a document, then they occur relatively frequent in that document. This distribution structure of good terms is called burstiness (occurrence is bursty in some documents). We also refer to this nature of terms as clinginess, meaning that the occurrences of good terms tend to cling together in the same document[Church and Gale, 1995]. An IDF-value reflects clinginess, and ILF-value does not. For example, let two terms t_1 and t_2 both occur in 100 locations. We have $ilf(t_1,c) = ilf(t_2,c)$. If the locations of t_1 are primarily in the same documents whereas locations of t_2 are distributed among many documents, then we have $idf(t_1,c) > idf(t_2,c)$, i.e. t_1 is more discriminative. From that point of view, IDF supports the effect achieved by the within-document term frequency. Therefore, we tend to use IDF for document retrieval and IEF for element retrieval.

With context-specific frequencies, we mainly investigate their performance with TF-IDF and LM:

Definition 9 *TF-IDF Retrieval Function with Context-Specific Discriminativeness*

$$\begin{aligned} RSV_{ief}(d,q) &:= \sum_t tf(t,q) \cdot tf(t,d) \cdot ief(t, root(d)) \\ RSV_{idf}(d,q) &:= \sum_t tf(t,q) \cdot tf(t,d) \cdot idf(t, root(d)) \\ RSV(d,q) &:= \begin{cases} RSV_{ief}(d,q) \text{ if } d \text{ is an element} \\ RSV_{idf}(d,q) \text{ if } d \text{ is a document} \end{cases} \end{aligned}$$

In the definition of the RSV for elements, we replace the collection by the root of the sub-tree in which the retrieved element is located. Thus, the RSV is based on the frequencies with respect to this root. The definitions are generalizations of classical TF-IDF retrieval, in which $root(d)$ is a collection where the document or element situate.

The retrieval function of LM stays in the same form when context specific frequencies are applied, but d can be element, document or collection, and $root(d)$ will be chosen according to the type of d. Different type frequency and different context can be chosen for the smoothing.

$$RSV(d,q) = \sum_t \log(1 + \frac{\lambda\, p(t|d)}{(1-\lambda)(p(t|root(d))} \tag{5.13}$$

From the next section, we will empirically investigate the performance of context-specific frequencies and discriminativeness.

5.7 Experiments with Context-specific Frequencies

For the context-specific frequencies experiment, we set up two types of retrieval: document retrieval and element retrieval. To understand whether a collection concerns a common subject will influence the retrieval quality of models utilizing context-specific frequencies, we run the the document and element retrieval in two environments: 1) each sub collection has a common subject, only contains document from one journal or proceedings; 2) or each sub-collection has no common subject, i.e. its document come from different journal or proceeding.

This section is structured as follows: Section 5.7.1 describes experimental settings context-specific frequency based retrieval. Section 5.7.2 explains the the experiment methodology based on TF-IDF retrieval model and lists the results. In this section the test collection is the INEX original collection, where each sub-collection has a subject. Section 5.7.3 analysis the retrieval results from the collection organized by subject, try to identify the type of queries that can benefit from the context-specific frequencies. Section 5.7.4 runs the same experiment introduced in section 5.7.2, on the reorganized INEX collection, where each sub-collection has no subject, contains article from different journals or proceedings. Results are also listed in this section. Section 5.7.5 analysis the retrieval results from the collection which is not organized by subject. Section 5.7.6 presents the methodology we used for element retrieval, results and analysis are also shown in this section. Section 5.7.7 shows how context-specific frequencies apply to language modelling, and the results from both the subject organized collection and the non-subject organized collection.

5.7.1 Experimental Settings

To be able to utilize the structure information in the documents and collections to investigate the context-specific frequencies, we choose INEX [1] as our test collection, which provides a large XML document collection, query sets and relevance judgements. The INEX document collection contains the publications of the IEEE computer science journals, which are organized into directories. The highest level directory is journal, and each journal is grouped into sub-directories by year. The INEX collection can be viewed as a document tree. The retrieval process can be carried out within any sub-tree (Journal or Journal+Year) or within the whole INEX document tree.

The INEX document collection includes 18 journal/magazines covering 8 years. It has

[1] Please refer to http://inex.is.informatik.uni-duisburg.de/ for more information on INEX

500MB of structured documents, around 12,000 articles , 15 million retrievable objects (sections, paragraphs, etc), and 32.5 million terms. Our investigation is to use context-specific frequencies to rank the element without preference of element type, therefore we use INEX content-only (CO) queries from year 2002 to year 2005. There are 116 assessed CO queries overall that will be used in the later context-specific frequencies experiment,

In our experiment, we use TREC evaluation software to measure MAP and P@10 of each retrieval runs to indicate the retrieval quality. The TREC evaluation program computes precision and recall based on binary document relevant assessment, which can not be applied to INEX relevant assessment in a straight forward manner, as the INEX test collection is designed for XML document retrieval, and it assesses the relevance of an element in two dimensions: "specificity" and "exaustivity". Each of the dimension is judged at 3 level 0, 1, or 2 [Fuhr et al., 2003a]. Therefore we adapt the relevant assessment of INEX to the TREC format. The rule we use is:

> If an element has any aspect judged with value greater than 0 ($specificity > 0$ or $exaustivity > 0$), then this element is relevant.
>
> If any element of a document is relevant, then this document is relevant.

All the experiment runs will be evaluated based on this adapted assessment. We evaluate the top 1500 retrieved documents from each run as INEX does.

Currently, our context-specific experiments focus on TF-IDF and LM. In TF-IDF we apply context-specific IDF in retrieval model; in LM, we use context-specific frequencies to smooth the probability $P(t|d)$.

5.7.2 Document Retrieval with TF-IDF in the Multi-collections Organized by Subject

In document retrieval, there are tree types of retrieval strategies: 1) use global discriminativeness to rank the documents; 2) use local discriminativeness to rank the documents; 3) choose top promising collections, and rank the documents from these collections with their local RSV. In each strategy, discriminativeness based on different element type will be used.

Settings for document retrieval:

- Use global IDF, IEF, ILF to weight terms and documents.
- Use local IDF, IEF, ILF to weight terms and documents.

- Go into the most promising collections, and use local IDF, IEF, ILF to weight terms and documents. The promising collection choosing criteria:
  - use $\sum_{t \in q} idf(t,c)$ to choose promising collection ($QInf(q,c)$)
  - use $\sum_{t \in q} df(t,c)$ to choose promising collection ($QDF(q,c)$)

When the retrieval uses global IDF, the IDF will be computed on the whole document collection . In INEX the global IDFis computed on the term's document frequency over the whole INEX document collection, i.e. $idf(t,c_{INEX}) = -\log \frac{n_D(t,c_{INEX})}{N_D(c_{INEX})}$. When the retrieval uses local IDF, the IDF will be computed based on a given sub-collection . In INEX, the sub-collections can be a journal or a journal in a specific year, i.e. $idf(t,c_{journal}) = -\log \frac{n_D(t,c_{journal})}{N_D(C_{journal})}$. Therefore, one term can have several different IDF values according to the context.

When documents are retrieved from sub-collections, they need to be merged into one ranking list. We merge the documents purely according to their local RSV. By local RSV we mean that the RSV is calculated based on query term's local IDF or local informativeness. Here we give an example of result merging: given the unordered TF-IDF results from each sub-collection: $0.5(d_1,c_1)$, $0.7(d_2,c_1)$, $0.8(d_3,c_2)$, and $0.6(d_4,c_3)$, the result list will be $0.5(d_1,c_1)$, $0.6(d_4,c_3)$, $0.7(d_2,c_1)$ and $0.8(d_3,c_2)$. There is no collection statistical information involved.

When retrieval is carried out within the top promising sub-collections, the retrieval process will first choose some good sub-collections likely to have more relevant documents, either based on the query's informativeness in a particular collection QInf(q,c), or the number of documents containing the query terms QDF(t,c). Then the retrieval processes are carried on within these selected sub-collections. The retrieved documents from the selection sub-collection are merged the same way as what was introduced in the local IDF strategy. In top promising strategy experiments, we set one third as a cut-off to choose the promising sub-collections. For example there are totally 18 journals, then we allow 6 sub-collections as promising. Similarly, if the documents are grouped by year, there will be 8 sub-collections, then we will allow 3 sub-collections as top promising.

The retrieval function has been introduced in section 5.6, we will not reiterate here.

Although it is known that document frequency is better than word frequency in document retrieval [Church and Gale, 1995], we are interested in investigating what is the performance of element frequency, which locates in the middle of document and word frequency. When we use context-specific frequencies to estimate these inverse frequencies, how do they behave

retrieval function	global *idf*(t,INEX)		local *idf*(t,journal)		Top-6 QDF(q,journal)		Top-6 QInf(q,journal)	
	MAP	P@10	MAP	P@10	MAP	P@10	MAP	P@10
idf_{doc}, tf_{max}	0.2217	0.4267	0.1767	0.3241	0.1637	0.3371	0.0203	0.0931
idf_{doc}, tf_{sum}	0.2019	0.3603	0.1737	0.3078	0.1654	0.3319	0.0230	***0.1276***
idf_{doc}, $tf_{Poissona}$	***0.3107***	0.5009	0.1382	0.2345	0.1497	0.2940	0.0277	0.1164
idf_{sec},tf_{max}	0.2069	0.4233	0.1828	0.3690	0.1650	***0.3690***	0.0159	0.0767
idf_{sec},tf_{sum}	0.1736	0.3155	0.1654	0.3121	0.1528	0.3121	0.0145	0.0681
idf_{sec} $tf_{poissona}$	0.3070	***0.5043***	0.1705	0.2940	0.1788	0.3103	0.0265	0.1129
idf_{para},tf_{max}	0.1973	0.4138	0.1802	0.3759	0.1635	0.3665	0.0142	0.0655
idf_{para},tf_{sum}	0.1604	0.2957	0.1553	0.2862	0.1422	0.2862	0.0132	0.0586
idf_{para} $tf_{poissona}$	0.3039	0.5034	0.1792	0.2974	0.1904	0.3078	0.0243	0.1017
idf_{loc},tf_{max}	0.1839	0.3810	0.1769	0.3655	0.1513	0.3328	0.0164	0.0741
idf_{loc},tf_{sum}	0.1426	0.2655	0.1390	0.2552	0.1220	0.2414	0.0147	0.0603
idf_{loc} $tf_{poissona}$	0.2984	0.5009	***0.2297***	***0.3810***	***0.2109***	0.3603	***0.0290***	0.1267

Table 5.2: TF-IDF: Document retrieval with context-specific frequencies in collection organized by subject

differently. Therefore, we use TF-IDF based retrieval function in context-specific frequencies, with different combination of TF and IDF estimation. The inverse frequencies that we use in this section are: inverse document frequency, inverse section frequency, inverse paragraph frequency and inverse location frequency. Figure 5.2 lists the result for all TF-IDF runs. And for each strategy, we display the best performance with italic font.

Our result shows that:

- With global frequencies, global IDF outperforms global IEF and global ILF. When the size of the element is decreasing, the performance of its inverse frequency drops. It is not surprising as document frequency shows better the clinginess of the term.

- When comparing global and local inverse frequencies, global IDF, IEF and ILF all distinctly outperform respectively local IDF, IEF and ILF. This result supports the views that in distributed retrieval the central indexed system are more effective than the system with data indexed distributively [Hawking and Thistlewaite, 1999].

- With the top promising sub-collection strategy, we find that choosing collections by QDF(q,c) has better performance than choosing with QInf(q,c). This confirms the work of [Callan et al., 1995b], which extends TF-IDF to collection selection. They believed that the collection having more documents containing the query terms tends to be more about the query, then this collection may possess more relevant documents. Although QInf(q,c) is theoretically proved more efficient from cost point of view, it does not perform well. Be-

cause it chooses the sub-collections that have less documents containing the query terms, which gives a small chances having potential relevant documents. Thus, it is not surprising that the strategy choosing sub-collections by QInf(q,c) has a poor performance.

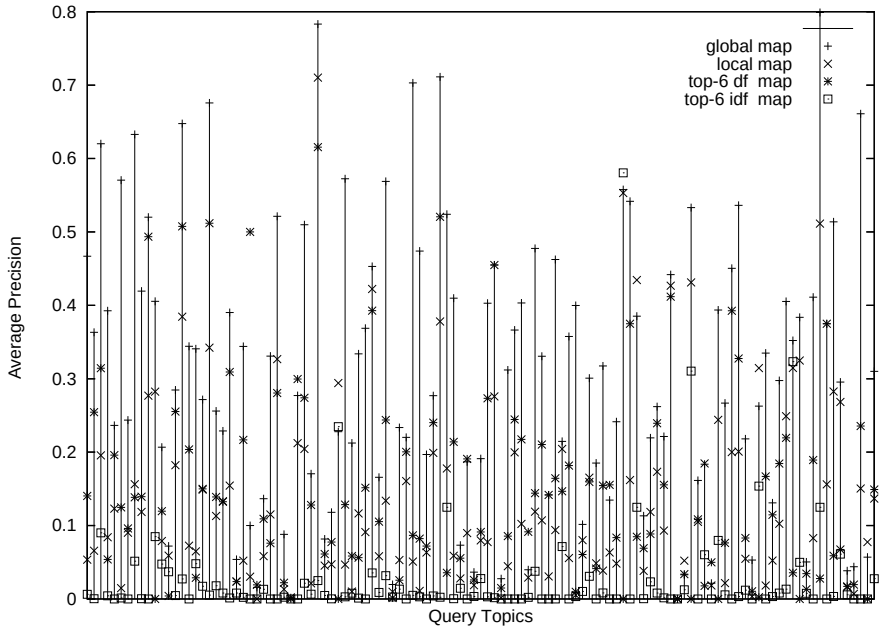
- Interestingly, the performances of local IDF and the top promising strategy with QDF(q,c) are very similar. In some runs, the top promising strategy with QDF(q,c) even performs better than local IDF strategy.

5.7.3 Analysis of TF-IDF Document Retrieval Results from the Multi-collections Organized by Subject

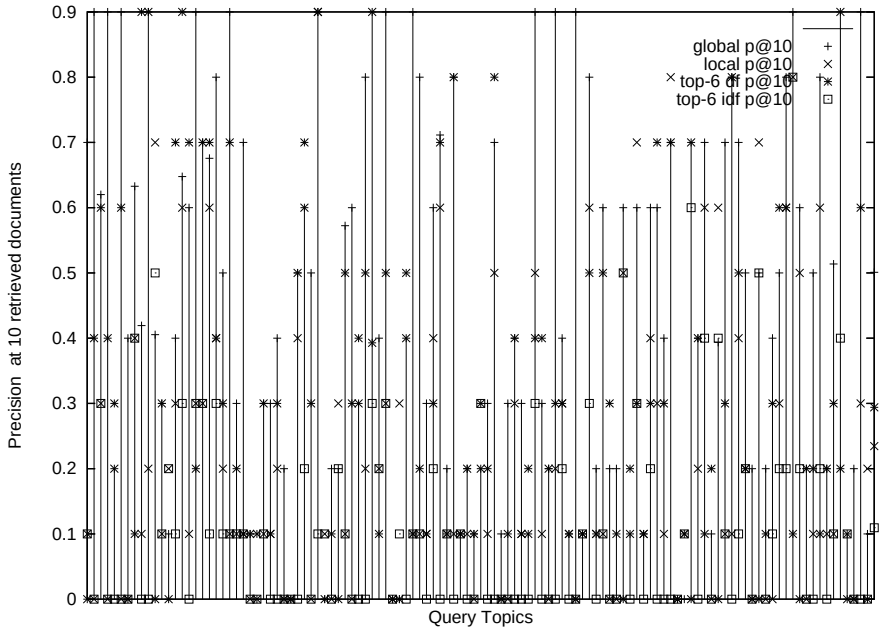
In this section we look into the details of each query, to see whether the context-specific frequency based model favors certain the query term distribution. As we observe that although some query terms are rare, but they appear in each sub-collection, while some query terms are frequent, but tend to clingy into few sub-collections. It would be useful to know whether such difference in distribution would impact the retrieval quality, thus, we can decide in which situation that context-specific frequency would be beneficial. Here, we mainly look at the retrieval function with Poisson based TF and document frequencies based IDF, which has the best MAP among all retrieval runs.

Table 5.2 shows the mean average precision (MAP) of different strategies, which are ranked with the order as global IDF, local IDF, top DF(QDF) and top IDF(QInf). We wonder if this is the case for any query. Therefore, we show each query's four strategies' AP and P@10 in figure 5.4. We found that there are some queries having better AP performance with top DF promising strategy than global IDF strategy, even more queries have better P@10 with top DF promising strategy than global IDF. Surprisingly there are few queries having higher AP and P@10 with top IDF promising strategy than global IDF, as the MAP and P@10 of top IDF promising strategy are far lower than global IDF. As these queries are only minority, so we will not look further into them.

We wonder whether the number of sub-collections in which the query terms occur correlates to the retrieval performances of local IDF strategy or top QDF promising strategy, therefore, we list all query term collection frequencies to observe. Table 5.3 shows top 5 and bottom 5 performed queries with local IDF promising strategy due the space issue. The top well performing queries with local IDF strategy, tend to have query terms occurring in all the sub-collections,



(a) MAP



(b) P@10

Figure 5.4: Comparison of single query performance results from global IDF, local IDF, top QDF(q,c) and top QInf(q,c) strategies in the multi-collection organized by subject

Qid	AP	P@10	Number of sub-collections containing query terms
112	0.6156	0.9000	cascad=18, styl=18, sheet=18, cont=18, scrambl=16, system=18
99	0.5207	0.7000	perl=18, featur=18
50	0.5118	0.7000	xml=18, edit=18, pars=18
46	0.5076	0.9000	firewal=16, internet=18, secur=18
100	0.5000	0.1000	associ=18, min=18, rul=18, med=18
⋮			
209	0.0000	0.0000	min=18, frequ=18, pattern=18, itemset=5, sequ=18, graph=18, associ=18
205	0.0000	0.0000	marshal=17, mcluhan=7
202	0.0000	0.0000	cas=18, stud=18
192	0.0000	0.0000	cybersick=1, nausea=7, virtu=18, real=18
117	0.0000	0.0000	patricia=14, tri=18

Table 5.3: CF statistics: Top 5 and bottom 5 queries with local IDF strategy

and the bottom well performing queries tend to have one or two query terms occurring in less sub-collections. Due to the small amount of sub-collections, and small amount of queries we use, we can not draw such conclusion safely. To better understand the impact of query term distribution, in table 5.4 we show the details of how many documents containing the query terms in each sub-collection. It shows that the query terms of the well performing queries occur in all the sub-collections, while the bad queries have some query terms which occur in few sub-collections, and have low DF value.

Table 5.3 and table 5.4 show that the query terms should be distributed into every collection in order to have a better retrieval result with local IDF strategy, but it does not necessarily lead to better result. For those queries with the worst AP and P@10 in local IDF strategy, they usually have 1 or 2 query terms that appear in very few sub-collections, and also rarely in these sub-collections. These rare terms will have high local IDF value, consequently they lead to high RSV for the documents containing these terms. If these documents are not relevant, then such a retrieval strategy will greatly damage the retrieval performance. This is the reason that some queries have bad retrieval performance with local IDF strategy.

Next we list all the queries that have better performances with local IDF than global IDF in table 5.5 and table 5.6, and queries having better performances with top QDF promising strategy than global IDF in table 5.7 and table 5.8. Queries that have very small difference of precision from two retrieval strategies, e.g. less than 0.01, will not be listed in the tables. In the following tables, the subscriptions G, L, T after AP and P@10 means global, local and top QDF promising strategy respectively.

Qid	AP_G	AP_L	Number of sub-collections containing the query terms
117	0.2278	0.2941	patricia=14, tri=18
218	0.2628	0.3145	comput=18, assist=18, compos=18, mus=18, not=18, midi=10

194	0.3852	0.4345	multi=18, lay=18, perceptron=12, radi=18, basi=18, funct=18, comparison=18
241	0.0570	0.0776	singl=18, sign=18, ldap=10
203	0.0330	0.0521	cod=18, sign=18, verifi=18
Qid	AP_G	AP_L	DF's in sub-collections
117	0.2278	0.2941	patricia=3,2,30,2,,1,4,9,3,5,4,14,,,,2,1,2, tri=134,150,370,120,76,216,100,48,126,104,73,265,197,185,58,166,317,190,
218	0.2628	0.3145	comput=316,680,1902,571,539,677,547,204,554,417,358,777,1042,756,225,571,1030,553, assist=82,109,216,61,48,157,60,28,54,59,34,98,82,41,59,89,77,111, compos=53,155,273,83,73,131,105,19,80,121,85,115,287,211,84,223,326,284, mus=42,67,104,21,10,51,44,12,32,111,14,41,5,3,5,19,25,21, not=301,597,1756,545,462,646,507,241,542,387,308,909,1000,745,202,553,998,538, midi=,7,10,3,1,4,,,2,22,,3,1,,,,1,,
194	0.3852	0.4345	multi=12,64,98,41,38,92,103,10,52,45,36,41,191,175,56,145,258,89, lay=74,186,455,148,118,196,212,69,191,141,112,163,139,135,64,127,186,162, perceptron=2,2,6,6,,12,,,4,,,1,4,2,,10,78,2, radi=30,73,49,89,31,56,2,4,27,14,9,6,27,4,41,15,173,10, basi=142,177,623,178,148,256,188,78,155,131,96,322,323,251,105,292,481,327, funct=184,389,953,356,313,407,306,131,374,237,201,483,793,579,183,482,879,452, comparison=74,159,269,145,141,162,84,27,149,68,104,160,596,457,135,365,648,293,
241	0.0570	0.0776	singl=154,407,939,324,306,380,310,142,391,244,237,450,793,614,166,461,756,442, sign=156,240,711,182,388,271,186,74,329,209,116,212,481,250,96,139,591,189, ldap=,1,18,3,,,34,10,,1,2,1,,1,,1,,,
203	0.0330	0.0521	cod=154,256,1090,339,199,276,283,128,321,213,209,638,493,338,97,240,402,389, sign=156,240,711,182,388,271,186,74,329,209,116,212,481,250,96,139,591,189, verifi=27,41,326,48,269,108,60,23,105,45,42,153,328,206,35,151,286,280,

Table 5.5: CF and DF statistics: Queries with better AP by local IDF strategy than global IDF strategy

Qid	$P@10_G$	$P@10_L$	Number of sub-collections containing the query terms
93	0.0000	0.3000	charl=18, babbag=8, institut=18, inst=18
42	0.4055	0.7000	decrypt=17, enigma=10, cod=18
209	0.3936	0.6000	min=18, frequ=18, pattern=18, itemset=5, sequ=18, graph=18, associ=18
218	0.5000	0.7000	comput=18, assist=18, compos=18, mus=18, not=18, midi=10
113	0.0000	0.1000	markov=17, model=18, user=18, behaviour=17
117	0.2000	0.3000	patricia=14, tri=18
194	0.6000	0.7000	multi=18, lay=18, perceptron=12, radi=18, basi=18, funct=18, comparison=18
201	0.7000	0.8000	web=18, www=18, relevanc=18, scor=18, rank=18
203	0.0000	0.1000	cod=18, sign=18, verifi=18
241	0.1000	0.2000	singl=18, sign=18, ldap=10
44	0.1000	0.2000	internet=18, soci=18, communic=18, netizen=7, soci=18, sociolog=5, web=18, usenet=16, mail=18, network=18, cultur=18
Qid	$P@10_G$	$P@10_L$	DF's in sub-collections
93	0.0000	0.3000	charl=166,31,178,47,34,51,59,16,19,26,25,54,31,50,10,12,30,15, babbag=136,,14,,2,1,1,,4,,,5,,,,1,,, institut=217,281,633,243,193,373,173,63,192,168,124,371,475,444,141,295,526,304, inst=34,38,147,88,12,154,18,9,22,34,63,106,123,109,47,82,180,103,
42	0.4055	0.7000	decrypt=15,6,74,2,7,5,26,10,15,12,4,8,16,4,,3,1,13, enigma=26,2,2,1,,6,1,,3,,3,2,,,,,,1, cod=154,256,1090,339,199,276,283,128,321,213,209,638,493,338,97,240,402,389,
209	0.3936	0.6000	min=90,121,312,111,68,252,70,38,113,56,102,147,392,388,87,299,408,209, frequ=103,116,473,93,72,180,111,61,127,77,85,252,260,242,54,220,209,228, pattern=100,274,541,184,209,314,94,54,166,127,124,259,469,402,117,318,1046,259, itemset=,,5,,,1,,,,,1,,,,,17,1,, sequ=96,217,385,156,152,226,101,24,152,151,112,154,586,491,135,325,615,347, graph=81,677,709,259,183,247,173,53,225,292,166,195,526,552,225,337,683,318, associ=232,374,1000,336,312,466,286,145,335,260,228,488,715,629,175,508,743,473,
218	0.5000	0.7000	comput=316,680,1902,571,539,677,547,204,554,417,358,777,1042,756,225,571,1030,553, assist=82,109,216,61,48,157,60,28,54,59,34,98,82,41,59,89,77,111, compos=53,155,273,83,73,131,105,19,80,121,85,115,287,211,84,223,326,284, mus=42,67,104,21,10,51,44,12,32,111,14,41,5,3,5,19,25,21,

			not=301,597,1756,545,462,646,507,241,542,387,308,909,1000,745,202,553,998,538, midi=,7,10,3,1,4,,,2,22,,3,1,,,,1,,,
113	0.0000	0.1000	markov=3,6,24,25,6,31,4,,10,7,2,9,99,43,3,33,233,42, model=196,555,1227,441,392,557,369,143,335,325,305,644,741,663,207,518,921,514, user=175,500,1380,305,256,496,460,195,405,390,268,617,288,275,163,423,262,389, behaviour=2,3,3,4,1,10,1,,2,4,3,2,12,5,8,16,24,25,
117	0.2000	0.3000	patricia=3,2,30,2,,1,4,9,3,5,4,14,,,,2,1,2, tri=134,150,370,120,76,216,100,48,126,104,73,265,197,185,58,166,317,190,
194	0.6000	0.7000	multi=12,64,98,41,38,92,103,10,52,45,36,41,191,175,56,145,258,89, lay=74,186,455,148,118,196,212,69,191,141,112,163,139,135,64,127,186,162, perceptron=2,2,6,6,,12,,,4,,,1,4,2,,10,78,2, radi=30,73,49,89,31,56,2,4,27,14,9,6,27,4,41,15,173,10, basi=142,177,623,178,148,256,188,78,155,131,96,322,323,251,105,292,481,327, funct=184,389,953,356,313,407,306,131,374,237,201,483,793,579,183,482,879,452, comparison=74,159,269,145,141,162,84,27,149,68,104,160,596,457,135,365,648,293,
201	0.7000	0.8000	web=86,312,989,222,121,346,464,193,197,312,151,349,93,95,45,144,124,112, www=64,334,928,293,164,443,458,172,256,264,201,333,99,103,63,119,154,154, relevanc=36,19,72,28,11,80,28,5,6,28,16,51,13,18,11,87,61,54, scor=18,40,87,29,7,92,25,14,16,49,2,61,10,14,6,53,129,38, rank=47,34,136,39,17,94,42,20,16,33,18,85,87,132,19,118,242,79,
203	0.0000	0.1000	cod=154,256,1090,339,199,276,283,128,321,213,209,638,493,338,97,240,402,389, sign=156,240,711,182,388,271,186,74,329,209,116,212,481,250,96,139,591,189, verifi=27,41,326,48,269,108,60,23,105,45,42,153,328,206,35,151,286,280,
241	0.1000	0.2000	singl=154,407,939,324,306,380,310,142,391,244,237,450,793,614,166,461,756,442, sign=156,240,711,182,388,271,186,74,329,209,116,212,481,250,96,139,591,189, ldap=,1,18,3,,,34,10,,1,2,1,,1,,1,,,
44	0.1000	0.2000	internet=74,184,972,140,56,257,547,183,218,280,151,267,57,84,19,102,33,89, soci=163,70,317,58,23,153,88,25,62,84,34,176,25,24,11,52,49,70, communic=53,156,496,90,67,206,212,82,143,148,139,250,180,327,21,100,28,216, netizen=3,,3,,,1,5,,1,3,1,,,,,,,, soci=163,70,317,58,23,153,88,25,62,84,34,176,25,24,11,52,49,70, sociolog=3,1,1,,,1,,,,1,,,,,,,,, web=86,312,989,222,121,346,464,193,197,312,151,349,93,95,45,144,124,112, usenet=8,3,26,2,1,9,24,1,5,2,5,11,4,,,3,2,3, mail=109,120,672,172,87,151,251,129,108,166,77,164,954,680,201,546,978,520, network=140,309,1236,256,187,483,453,184,353,348,298,367,650,664,66,364,422,323, cultur=93,56,209,38,22,73,50,30,23,79,9,234,6,10,3,14,22,15,

Table 5.6: CF and DF Statistics: Queries with better P@10 by local IDF strategy than global IDF strategy

Qid	AP_G	AP_T	Number of sub-collections containing the query terms
100	0.1000	0.5000	associ=18, min=18, rul=18, med=18
174	0.0398	0.0915	internet=18, web=18, pag=18, prefetch=17, algorithm=18, cpu=18, mem=18, disk=18
208	0.0214	0.0500	artifici=18, intellig=18, hist=18
109	0.2771	0.2996	cpu=18, cool=18, cool=18, fan=18, design=18, design=18, heat=18, dissip=18, airflow=12, cas=18
190	0.1347	0.1555	commerc=18, bus=18, data=18, warehous=18
Qid	AP_G	AP_T	DF's in sub-collections
100	0.1000	0.5000	associ=232,374,1000,336,312,466,286,145,335,260,228,488,715,629,175,508,743,473, min=90,121,312,111,68,252,70,38,113,56,102,147,392,388,87,299,408,209, rul=106,147,510,139,133,348,169,93,188,111,91,310,279,221,74,387,433,323, med=37,34,88,20,11,38,9,4,18,13,8,28,47,60,18,35,164,49,
174	0.0398	0.0915	internet=74,184,972,140,56,257,547,183,218,280,151,267,57,84,19,102,33,89, web=86,312,989,222,121,346,464,193,197,312,151,349,93,95,45,144,124,112, pag=163,198,687,152,128,229,327,106,202,196,134,270,133,125,44,183,118,145, prefetch=2,4,52,3,7,2,8,,53,9,25,2,79,36,6,25,1,13, algorithm=82,380,729,392,263,401,186,43,251,181,238,188,894,727,203,487,943,385, cpu=26,82,234,93,100,33,67,13,196,43,85,39,203,149,53,127,69,98, mem=86,21,134,34,165,41,12,1,115,20,46,10,159,98,11,30,7,36, disk=81,132,303,91,40,57,70,37,123,105,88,70,101,84,37,182,81,57,

208	0.0214	0.0500	artifici=63,129,341,87,25,479,81,10,57,83,52,82,88,90,47,294,557,120, intellig=100,150,574,101,68,663,216,55,143,170,106,155,126,132,64,372,1046,133, hist=64,6,35,11,1,21,16,5,3,6,11,17,12,8,2,37,10,29,
109	0.2771	0.2996	cpu=26,82,234,93,100,33,67,13,196,43,85,39,203,149,53,127,69,98, cool=32,31,64,30,28,30,16,10,39,8,15,23,16,16,6,10,16,8, cool=32,31,64,30,28,30,16,10,39,8,15,23,16,16,6,10,16,8, fan=23,37,73,15,42,29,18,5,42,21,14,39,86,45,10,26,39,17, design=264,549,1596,403,539,587,425,175,519,374,302,788,921,690,189,480,621,500, design=264,549,1596,403,539,587,425,175,519,374,302,788,921,690,189,480,621,500, heat=42,36,96,86,33,55,16,5,42,5,12,26,11,11,13,9,29,8, dissip=10,7,34,27,66,5,2,2,79,2,3,4,36,10,5,3,5,2, airflow=1,6,3,7,,3,1,,4,,2,1,,1,2,1,,, cas=203,428,1115,380,355,487,312,158,385,254,250,695,965,727,200,538,948,521,
190	0.1347	0.1555	commerc=32,40,339,19,12,101,194,129,66,90,41,94,25,23,4,46,13,39, bus=209,178,1007,134,249,297,279,219,403,186,162,561,294,278,19,146,56,190, data=221,529,1475,420,377,558,437,213,447,373,302,657,756,641,196,585,880,486, warehous=6,11,80,6,1,39,20,32,9,6,14,24,4,3,4,73,2,5,

Table 5.7: CF and DF statistics: Queries with better AP by top QDF promising strategy than global IDF strategy

Qid	$P@10_G$	$P@10_T$	number of sub-collection containing the query term
40	0.4194	0.9000	cont=18, bas=18, retriev=18
45	0.4000	0.7000	augm=18, real=18, medicin=18
46	0.6476	0.9000	firewal=16, internet=18, secur=18
101	0.0000	0.1000	test=18, inform=18
110	0.6000	0.7000	stream=18, deliv=18, stream=18, synchroniz=18, audio=18, video=18, stream=18, appli=18
123	0.3000	0.4000	multidimension=18, ind=18, near=18, neighbour=11, search=18
167	0.2000	0.3000	que=18, proces=18, spati=18, data=18, multimedia=18, min=18
169	0.7000	0.8000	que=18, expans=18, relevanc=18, feedback=18, web=18
174	0.1000	0.2000	internet=18, web=18, pag=18, prefetch=17, algorithm=18, cpu=18, mem=18, disk=18
176	0.3000	0.4000	secur=18, web=18, cook=18, authenti=17, integr=18, confidential=17
190	0.2000	0.3000	commerc=18, bus=18, data=18, warehous=18
193	0.1000	0.2000	good=18, tur=18, estim=18, smooth=18
198	0.6000	0.7000	appli=18, develop=18, python=15, java=18, comparison=18
203	0.0000	0.1000	cod=18, sign=18, verifi=18
208	0.1000	0.2000	artifici=18, intellig=18, hist=18
222	0.5000	0.6000	eletron=3, commerc=18, bus=18, strateg=18
36	0.2000	0.3000	heat=18, dissip=18, microcomput=18, chip=18
47	0.6000	0.7000	concurren=18, control=18, semant=18, transact=18, manag=18, appli=18, performanc=18, benefit=18
94	0.4000	0.5000	hyperlink=18, analysi=18, distil=18
50	0.6757	0.7000	xml=18, edit=18, pars=18
Qid	$P@10_G$	$P@10_T$	DF's in sub-collections
40	0.4194	0.9000	cont=108,226,662,123,119,279,304,107,198,296,95,254,212,177,61,253,230,229, bas=252,609,1664,475,468,650,489,226,517,421,319,798,963,732,208,561,989,538, retriev=51,97,351,67,15,245,154,37,58,177,60,93,84,87,41,325,197,122,
45	0.4000	0.7000	augm=24,94,157,39,36,101,43,11,50,65,36,52,125,110,39,128,113,103, real=210,574,1307,390,274,534,363,183,406,367,267,695,549,451,178,426,798,442, medicin=35,84,106,67,2,94,15,9,10,33,16,35,7,10,37,49,81,18,
46	0.6476	0.9000	firewal=,8,98,6,4,3,72,46,9,12,13,32,5,5,,3,1,6, internet=74,184,972,140,56,257,547,183,218,280,151,267,57,84,19,102,33,89, secur=98,72,761,80,42,140,327,165,120,123,122,242,136,70,4,117,41,138,
101	0.0000	0.1000	test=156,308,1027,277,539,401,211,107,296,172,175,615,518,334,151,351,750,424, inform=283,556,1637,428,392,654,502,237,445,427,311,771,963,729,204,572,1022,545,
110	0.6000	0.7000	stream=32,120,315,58,56,85,128,39,130,190,86,55,182,115,37,81,115,81, deliv=108,121,511,81,108,140,201,107,150,168,99,308,152,215,15,75,33,155, stream=32,120,315,58,56,85,128,39,130,190,86,55,182,115,37,81,115,81, synchroniz=16,77,248,58,71,42,96,25,114,139,141,64,233,330,18,93,22,175, audio=45,137,265,29,32,59,96,27,98,275,41,31,34,15,7,55,37,30, video=61,303,497,60,70,126,143,54,168,372,75,87,80,62,44,105,248,41,

			stream=32,120,315,58,56,85,128,39,130,190,86,55,182,115,37,81,115,81, appli=227,680,1625,499,444,615,474,212,503,419,343,741,939,713,207,550,973,529,
123	0.3000	0.4000	multidimension=3,62,68,43,10,20,5,12,14,18,16,13,47,114,48,86,122,28, ind=104,202,480,176,97,253,182,67,158,192,124,172,811,618,167,494,819,410, near=154,311,618,258,144,253,179,84,205,139,109,262,341,314,152,181,603,169, neighbour=1,1,,1,,1,,,,,1,,2,1,2,2,15,1, search=115,177,541,158,91,397,211,86,108,170,110,196,334,277,109,355,592,221,
167	0.2000	0.3000	que=32,87,348,37,4,189,161,53,29,109,53,94,50,82,45,395,86,116, proces=251,579,1626,442,456,606,416,205,534,383,333,820,926,746,204,549,972,524, spati=7,238,161,138,20,97,19,1,44,111,43,15,89,99,125,146,478,36, data=221,529,1475,420,377,558,437,213,447,373,302,657,756,641,196,585,880,486, multimedia=21,201,530,32,71,108,148,41,152,465,106,87,123,127,40,225,90,69, min=90,121,312,111,68,252,70,38,113,56,102,147,392,388,87,299,408,209,
169	0.7000	0.8000	que=32,87,348,37,4,189,161,53,29,109,53,94,50,82,45,395,86,116, expans=54,41,122,77,31,53,26,16,49,14,30,40,166,82,29,76,185,50, relevanc=36,19,72,28,11,80,28,5,6,28,16,51,13,18,11,87,61,54, feedback=26,153,288,75,108,183,69,29,88,98,48,205,151,57,31,94,74,118, web=86,312,989,222,121,346,464,193,197,312,151,349,93,95,45,144,124,112,
174	0.1000	0.2000	internet=74,184,972,140,56,257,547,183,218,280,151,267,57,84,19,102,33,89, web=86,312,989,222,121,346,464,193,197,312,151,349,93,95,45,144,124,112, pag=163,198,687,152,128,229,327,106,202,196,134,270,133,125,44,183,118,145, prefetch=2,4,52,3,7,2,8,,53,9,25,2,79,36,6,25,1,13, algorithm=82,380,729,392,263,401,186,43,251,181,238,188,894,727,203,487,943,385, cpu=26,82,234,93,100,33,67,13,196,43,85,39,203,149,53,127,69,98, mem=86,21,134,34,165,41,12,1,115,20,46,10,159,98,11,30,7,36, disk=81,132,303,91,40,57,70,37,123,105,88,70,101,84,37,182,81,57,
176	0.3000	0.4000	secur=98,72,761,80,42,140,327,165,120,123,122,242,136,70,4,117,41,138, web=86,312,989,222,121,346,464,193,197,312,151,349,93,95,45,144,124,112, cook=19,31,64,12,3,31,32,14,8,11,12,35,12,11,25,23,7,34, authenti=,12,161,10,1,21,110,37,26,25,21,34,15,12,1,12,24,26, integr=156,375,1134,319,394,458,326,176,355,299,196,516,454,303,143,374,535,371, confidential=5,4,59,4,2,4,26,12,10,8,5,20,3,3,,10,3,19,
190	0.2000	0.3000	commerc=32,40,339,19,12,101,194,129,66,90,41,94,25,23,4,46,13,39, bus=209,178,1007,134,249,297,279,219,403,186,162,561,294,278,19,146,56,190, data=221,529,1475,420,377,558,437,213,447,373,302,657,756,641,196,585,880,486, warehous=6,11,80,6,1,39,20,32,9,6,14,24,4,3,4,73,2,5,
193	0.1000	0.2000	good=212,288,848,319,247,388,270,154,313,177,171,637,474,399,130,320,677,315, tur=73,6,58,7,3,33,12,1,6,1,7,17,10,5,1,15,11,16, estim=84,145,450,194,169,200,95,76,138,87,80,352,414,291,106,248,824,248, smooth=38,189,142,104,23,89,37,22,44,73,34,78,47,42,134,56,518,47,
198	0.6000	0.7000	appli=227,680,1625,499,444,615,474,212,503,419,343,741,939,713,207,550,973,529, develop=294,575,1723,475,452,625,479,215,514,414,312,897,727,600,180,494,779,505, python=1,3,18,24,,1,14,2,1,2,4,6,,1,3,1,,2, java=5,85,426,77,27,92,280,72,90,87,78,174,22,28,9,34,9,91, comparison=74,159,269,145,141,162,84,27,149,68,104,160,596,457,135,365,648,293,
203	0.0000	0.1000	cod=154,256,1090,339,199,276,283,128,321,213,209,638,493,338,97,240,402,389, sign=156,240,711,182,388,271,186,74,329,209,116,212,481,250,96,139,591,189, verifi=27,41,326,48,269,108,60,23,105,45,42,153,328,206,35,151,286,280,
208	0.1000	0.2000	artifici=63,129,341,87,25,479,81,10,57,83,52,82,88,90,47,294,557,120, intellig=100,150,574,101,68,663,216,55,143,170,106,155,126,132,64,372,1046,133, hist=64,6,35,11,1,21,16,5,3,6,11,17,12,8,2,37,10,29,
222	0.5000	0.6000	eletron=1,,,,,,,,,1,,1,,,,,, commerc=32,40,339,19,12,101,194,129,66,90,41,94,25,23,4,46,13,39, bus=209,178,1007,134,249,297,279,219,403,186,162,561,294,278,19,146,56,190, strateg=98,146,689,182,238,317,168,140,170,125,172,403,420,411,95,349,356,291,
36	0.2000	0.3000	heat=42,36,96,86,33,55,16,5,42,5,12,26,11,11,13,9,29,8, dissip=10,7,34,27,66,5,2,2,79,2,3,4,36,10,5,3,5,2, microcomput=44,10,53,5,7,12,5,3,47,6,2,10,19,13,2,6,3,6, chip=43,77,489,58,365,60,48,21,374,49,64,55,324,148,9,15,33,27,
47	0.6000	0.7000	concurren=3,2,131,24,32,24,43,3,23,14,284,44,103,147,1,118,5,142, control=218,458,1251,310,335,506,372,151,426,287,234,608,664,514,167,381,528,467, semant=15,45,304,33,45,187,135,14,46,108,76,120,92,108,23,317,93,294,

			transact=57,46,446,52,58,114,190,103,119,57,119,155,1042,765,219,584,1044,570, manag=213,299,1408,237,267,452,416,215,344,295,267,755,345,345,75,479,151,390, appli=227,680,1625,499,444,615,474,212,503,419,343,741,939,713,207,550,973,529, performanc=109,321,1105,333,331,363,262,140,410,200,293,444,734,660,135,432,728,356, benefit=101,214,744,154,173,255,208,126,227,143,135,459,271,242,82,223,164,300,
94	0.4000	0.5000	hyperlink=1,26,55,11,2,34,47,6,5,47,3,11,1,1,2,14,3,3, analysi=169,323,908,368,303,429,169,97,235,186,218,581,761,614,179,455,1045,498, distil=4,10,18,13,2,12,14,3,8,9,4,24,1,1,1,4,1,18,
50	0.6757	0.7000	xml=1,13,146,10,7,57,149,53,9,37,9,19,1,2,2,16,2,9, edit=254,287,781,320,215,320,244,79,199,248,159,427,329,317,91,262,310,310, pars=7,24,123,19,12,84,81,17,17,38,26,49,30,27,8,64,38,87,

Table 5.8: CF and DF statistics: queries with better P@10 by top QDF promising strategy than global IDF strategy

We observe that in 116 queries, there are 7 queries which have better AP with local IDF than global IDF and 31 queries which have better P@10 with local IDF than global IDF; there are 9 queries which have better AP with top QDF promising than global IDF, and 46 queries which have better P@10 with top QDF promising than global IDF. We didn't show all the queries which have better performance with local IDF or top QDF strategy in tables 5.5,5.6, 5.7 and 5.8, only those queries with the difference of AP or P@10 from two strategies greater than or equal to 0.01.

Queries having better performance with local IDF strategy always have one or two query terms occurring in few sub-collections and very rarely. While the queries having better performance with top QDF promising strategy always have query terms occurring in all the sub-collections. This can be the reason that top QDF promising strategy works well with such kind of queries. It does not matter which sub-collections are chosen, there should always be some relevant documents existing in these sub-collections. On the other hand, if the query terms only occur in few sub-collections and these sub-collections are not chosen as promising sub-collection, then there is a high chance that retrieved documents are not relevant. As a result, extremely low AP for these queries with the top QDF promising strategy.

5.7.4 Document Retrieval with TF-IDF in Multi-collections Organized without Subject

In the previous section 5.7.2, the test collection is organized by the subjects, where some terms tend to appear more in the sub-collection than others when the sub-collection concerning related subject. We postulate that such term distribution will make the context-specific retrieval having different performance to that from the test collection whose sub-collection has no common subject. In the latter case, the term distributions in each sub-collection would be similar.

In order to find out the impact of the different term distributions in the sub-collections on the

Qid	AP	P@10	Df in each sub-collection
112	0.6156	0.9000	cascad=4,9,55,19,17,17,35,11,32,11,3,9,81,29,7,29,38,17, styl=104,115,329,82,90,102,110,34,94,108,67,190,80,58,27,79,95,200, sheet=62,67,92,36,23,33,43,17,39,23,5,35,15,8,18,6,42,19, cont=108,226,662,123,119,279,304,107,198,296,95,254,212,177,61,253,230,229, scrambl=4,6,37,2,3,8,13,4,9,15,2,7,4,4,,1,2,, system=266,606,1780,494,504,696,508,234,559,436,357,875,970,765,198,565,935,548,
99	0.5207	0.7000	perl=2,6,65,25,7,9,52,15,15,8,3,27,19,5,1,6,1,13, featur=160,421,974,284,275,401,302,122,371,296,188,416,392,329,163,354,816,341,
50	0.5118	0.7000	xml=1,13,146,10,7,57,149,53,9,37,9,19,1,2,2,16,2,9, edit=254,287,781,320,215,320,244,79,199,248,159,427,329,317,91,262,310,310, pars=7,24,123,19,12,84,81,17,17,38,26,49,30,27,8,64,38,87,
46	0.5076	0.9000	firewal=,8,98,6,4,3,72,46,9,12,13,32,5,5,,3,1,6, internet=74,184,972,140,56,257,547,183,218,280,151,267,57,84,19,102,33,89, secur=98,72,761,80,42,140,327,165,120,123,122,242,136,70,4,117,41,138,
100	0.5000	0.1000	associ=232,374,1000,336,312,466,286,145,335,260,228,488,715,629,175,508,743,473, min=90,121,312,111,68,252,70,38,113,56,102,147,392,388,87,299,408,209, rul=106,147,510,139,133,348,169,93,188,111,91,310,279,221,74,387,433,323, med=37,34,88,20,11,38,9,4,18,13,8,28,47,60,18,35,164,49,
⋮			
209	0.0000	0.0000	min=90,121,312,111,68,252,70,38,113,56,102,147,392,388,87,299,408,209, frequ=103,116,473,93,72,180,111,61,127,77,85,252,260,242,54,220,209,228, pattern=100,274,541,184,209,314,94,54,166,127,124,259,469,402,117,318,1046,259, itemset=,,5,,,1,,,,,1,,,,,17,1,, sequ=96,217,385,156,152,226,101,24,152,151,112,154,586,491,135,325,615,347, graph=81,677,709,259,183,247,173,53,225,292,166,195,526,552,225,337,683,318, associ=232,374,1000,336,312,466,286,145,335,260,228,488,715,629,175,508,743,473,
205	0.0000	0.0000	marshal=15,4,26,5,,15,6,2,5,5,5,5,8,7,3,11,7,22, mcluhan=1,,2,1,,1,,,5,5,1,,,,,,,,
202	0.0000	0.0000	cas=203,428,1115,380,355,487,312,158,385,254,250,695,965,727,200,538,948,521, stud=252,359,968,370,243,460,216,119,239,232,219,550,622,557,147,441,641,443,
192	0.0000	0.0000	cybersick=,4,,,,,,,,,,,,,,,, nausea=1,8,2,,,3,,,1,1,,1,,,,,,, virtu=105,425,719,187,121,198,257,91,213,268,179,219,251,315,126,163,191,163, real=210,574,1307,390,274,534,363,183,406,367,267,695,549,451,178,426,798,442,
117	0.0000	0.0000	patricia=3,2,30,2,,1,4,9,3,5,4,14,,,,2,1,2, tri=134,150,370,120,76,216,100,48,126,104,73,265,197,185,58,166,317,190,

Table 5.4: DF statistics in each sub-collection: Top 5 and bottom 5 queries with local IDF strategy

retrieval function	global idf(t,INEX)		local idf(t,year)		Top-3 QDF(q,year)		Top-3 QInf(q,year)	
	MAP	P@10	MAP	P@10	MAP	P@10	MAP	P@10
idf_{doc}, tf_{max}	0.2217	0.4267	0.2146	0.4115	0.1152	0.3517	0.0621	0.2345
idf_{doc}, tf_{sum}	0.2019	0.3603	0.1999	0.3638	0.1096	0.3060	0.0621	0.2345
idf_{doc}, $tf_{Poissona}$	***0.3107***	0.5009	0.2576	0.3733	0.1530	***0.4216***	***0.0951***	***0.3096***
idf_{sec},tf_{max}	0.2069	0.4233	0.2023	0.4121	0.1151	0.3353	0.0525	0.2172
idf_{sec},tf_{sum}	0.1736	0.3155	0.1725	0.3224	0.1046	0.2922	0.0438	0.1862
idf_{sec} $tf_{poissona}$	0.3070	***0.5043***	0.2656	0.3957	***0.1647***	0.4155	0.0623	0.2422
idf_{para},tf_{max}	0.1973	0.4138	0.1923	0.4052	0.1118	0.3284	0.0537	0.2060
idf_{para},tf_{sum}	0.1604	0.2957	0.1584	0.2940	0.0924	0.2655	0.0402	0.1793
idf_{para} $tf_{poissona}$	0.3039	0.5034	0.2621	0.3966	0.1613	0.4096	0.0643	0.2405
idf_{loc},tf_{max}	0.1839	0.3810	0.1801	0.3853	0.1043	0.3147	0.0478	0.2009
idf_{loc},tf_{sum}	0.1426	0.2655	0.1402	0.2664	0.0873	0.2517	0.0395	0.1543
idf_{loc} $tf_{poissona}$	0.2984	0.5009	***0.2708***	***0.4198***	0.1583	0.4147	0.0629	0.2397

Table 5.9: TF-IDF: Document retrieval with context-specific frequencies in collection organized without subject

retrieval quality, we reorganized the collection and run the same retrieval strategies described in section 5.7.2. The retrieval results from the reorganized collection are shown in table 5.9.

Similar to the context-specific frequencies experiment on the collection organized by subject, this experiment shows that globe IDF outperforms global IEF and global ILF . Global IDF , IEF and ILF all distinctly out perform respectively local IDF , IEF , ILF . However, the results from the collection organized without subject show that the difference of global IDF and local IDF are not as distinguishable as that from the collection organized by subject. Actually the performances of local IDF and global IDF are quite similar when the retrieval processes are carried on the collection organized without subject. This can be explained by the fact that the distribution of terms in the sub-collection is similar to the the distribution in the whole collection, because there is no common subject in the sub-collections. This result is similar to the work in [Church and Gale, 1995], which have showed that the term distributions of a journal in each year are stable to that of the next year. Therefore purely using the local IDF without normalization can achieve the similar result as global IDF.

The result that the local IDF strategy has a similar retrieval performance to global IDF, demonstrates that the distributed retrieval can have similar performance to centralized retrieval if documents are randomly distributed among the data collections. No global statistical information is needed in the broker of distributed retrieval system. This discovery enables the distributed retrieval system to avoid the resource intensive task of maintaining centralized statistics.

The results of top QDF promising strategy from non-subject oriented collection is not as

good as what we got from the collection organized by subject. The top QDF promising strategy's MAP is only half of the global IDF's. This is understandable as when the documents are grouped randomly, and there is no a common subject for each sub-collection, the relevant documents can be assumed randomly distributed in each sub-collection. As we only choose one third of the collections to retrieve, then two thirds of the relevant documents will be missing. However, the loss of mean average precision can boost the efficiency of the retrieval system.

Still the performance of top QInf(q,c) promising strategy is very low when the retrieval is carried on the sub-collections without subject. Hence, using the informativeness of the query within the sub-collection is not good criterion to choose promising sub-collections.

5.7.5 Analysis of TF-IDF Document Retrieval Results from Multi-collections Organized without Subject

Again we choose Poisson based TF and document based IDF retrieval function to compare its retrieval results from different strategies. Figure 5.5 shows that there are many queries that have better AP or P@10 with local IDF or top QDF promising strategy than global IDF. Out of 116 query topics, for local IDF strategy, there are 17 queries which have AP no less than global IDF, and 53 queries which have P@10 no less than global IDF; for top QDF promising strategy, there are 12 queries which have AP no less than global IDF, and 61 queries whose AP and P@10 are no less than global IDF. Still we only list those queries whose AP or P@10 difference are greater than or equal to 0.01 in tables 5.10,5.11, 5.12 and 5.13.

It shows the similar statistical results to the experiment in the collection organized by subject. For those queries having better retrieval quality with local IDF or top QDF promising strategy than global strategy, most of their query terms occur in almost all the sub-collections, also frequently in the sub-collections.

There are more queries having a better performance with the local IDF or top QDF promising strategy when queries are executed on the text collection organized without subject, especially so when looking into the precision at 10 retrieval document.

Interestingly, the well performing queries with local IDF or top QDF promising strategy on the collection organized without subject are not those well performing ones on the collection organized with subject. However, there is no remarkable observation showing which kind of queries suit collection with subject or without subject when using local IDF or top QDF promising strategy. The only conclusion we can draw is that more queries work better using local

IDF or top QDF promising strategy than global IDF strategy when the retrieval is carried on the collection without subject.

Qid	AP_G	AP_L	Number of sub-collections containing the query terms
104	0.5213	0.5966	toy=18, sto=18
192	0.5577	0.5811	cybersick=1, nausea=7, virtu=18, real=18
123	0.3339	0.3562	multidimension=18, ind=18, near=18, neighbour=11, search=18
229	0.0504	0.0699	lat=18, semant=18, anlysi=3, lat=18, semant=18, ind=18
213	0.5362	0.5542	gib=16, sampl=18
227	0.3521	0.3699	adaboost=2, bag=18, ensembl=17, learn=18
169	0.4550	0.4712	que=18, expans=18, relevanc=18, feedback=18, web=18
198	0.2621	0.2744	appli=18, develop=18, python=15, java=18, comparison=18
Qid	AP_G	AP_L	DF's in sub-collections
104	0.5213	0.5966	toy=26,29,48,11,5,41,10,5,23,7,13,18,7,4,8,20,39,20, sto=170,106,196,56,24,110,86,47,63,74,24,153,1,7,6,43,12,24,
192	0.5577	0.5811	cybersick=,4,,,,,,,,,,,,,,,, nausea=1,8,2,,,3,,,1,1,,1,,,,,,, virtu=105,425,719,187,121,198,257,91,213,268,179,219,251,315,126,163,191,163, real=210,574,1307,390,274,534,363,183,406,367,267,695,549,451,178,426,798,442,
123	0.3339	0.3562	multidimension=3,62,68,43,10,20,5,12,14,18,16,13,47,114,48,86,122,28, ind=104,202,480,176,97,253,182,67,158,192,124,172,811,618,167,494,819,410, near=154,311,618,258,144,253,179,84,205,139,109,262,341,314,152,181,603,169, neighbour=1,1,,1,,1,,,,,1,,2,1,2,2,15,1, search=115,177,541,158,91,397,211,86,108,170,110,196,334,277,109,355,592,221,
229	0.0504	0.0699	lat=260,320,946,301,267,370,268,124,340,231,158,541,496,481,100,357,457,360, semant=15,45,304,33,45,187,135,14,46,108,76,120,92,108,23,317,93,294, anlysi=,,,,1,,,,,,,,,,,1,2, lat=260,320,946,301,267,370,268,124,340,231,158,541,496,481,100,357,457,360, semant=15,45,304,33,45,187,135,14,46,108,76,120,92,108,23,317,93,294, ind=104,202,480,176,97,253,182,67,158,192,124,172,811,618,167,494,819,410,
213	0.5362	0.5542	gib=11,2,19,11,,10,9,3,,11,1,11,2,3,12,11,93,5, sampl=50,229,350,178,122,208,71,47,149,128,56,188,207,143,138,214,685,210,
227	0.3521	0.3699	adaboost=,,,,,1,,,,,,,,,,,4,, bag=13,7,26,7,9,23,8,3,8,4,6,18,4,5,4,17,22,15, ensembl=2,11,13,21,3,13,3,,4,10,13,3,9,20,3,6,51,7, learn=185,204,735,211,112,472,180,119,154,196,106,568,82,71,31,204,403,208,
169	0.4550	0.4712	que=32,87,348,37,4,189,161,53,29,109,53,94,50,82,45,395,86,116, expans=54,41,122,77,31,53,26,16,49,14,30,40,166,82,29,76,185,50, relevanc=36,19,72,28,11,80,28,5,6,28,16,51,13,18,11,87,61,54, feedback=26,153,288,75,108,183,69,29,88,98,48,205,151,57,31,94,74,118, web=86,312,989,222,121,346,464,193,197,312,151,349,93,95,45,144,124,112,
198	0.2621	0.2744	appli=227,680,1625,499,444,615,474,212,503,419,343,741,939,713,207,550,973,529, develop=294,575,1723,475,452,625,479,215,514,414,312,897,727,600,180,494,779,505, python=1,3,18,24,,1,14,2,1,2,4,6,,1,3,1,,2, java=5,85,426,77,27,92,280,72,90,87,78,174,22,28,9,34,9,91, comparison=74,159,269,145,141,162,84,27,149,68,104,160,596,457,135,365,648,293,

Table 5.10: CF and DF: Queries with better AP by local IDF strategy than global IDF in the collection organized without subject

Qid	$P@10_G$	$P@10_L$	Number of sub-collections containing the query terms
38	0.4000	0.7000	multidimension=18, ind=18
119	0.5724	0.8000	optimiz=18, join=18, rel=18, databas=18
50	0.6757	0.9000	xml=18, edit=18, pars=18
228	0.6000	0.8000	ipv6=11, deploy=18, ipv6=11, support=18
46	0.6476	0.8000	firewal=16, internet=18, secur=18
104	0.4000	0.5000	toy=18, sto=18
123	0.3000	0.4000	multidimension=18, ind=18, near=18, neighbour=11, search=18
176	0.3000	0.4000	secur=18, web=18, cook=18, authenti=17, integr=18, confidential=17

213	0.7000	0.8000	gib=16, sampl=18
Qid	$P@10_G$	$P@10_L$	DF's in sub-collections
38	0.4000	0.7000	multidimension=3,62,68,43,10,20,5,12,14,18,16,13,47,114,48,86,122,28, ind=104,202,480,176,97,253,182,67,158,192,124,172,811,618,167,494,819,410,
119	0.5724	0.8000	optimiz=43,189,495,245,242,208,113,57,234,91,155,174,485,363,130,328,501,228, join=167,126,350,82,81,108,123,40,90,82,61,156,356,349,89,298,334,218, rel=237,507,1341,410,374,568,407,201,396,320,274,695,846,679,192,543,939,521, databas=59,203,780,146,69,357,278,135,103,221,158,340,157,190,44,493,354,231,
50	0.6757	0.9000	xml=1,13,146,10,7,57,149,53,9,37,9,19,1,2,2,16,2,9, edit=254,287,781,320,215,320,244,79,199,248,159,427,329,317,91,262,310,310, pars=7,24,123,19,12,84,81,17,17,38,26,49,30,27,8,64,38,87,
228	0.6000	0.8000	ipv6=,1,39,3,,,50,13,5,6,2,2,1,2,,,,, deploy=15,53,379,33,46,130,217,105,60,91,67,171,35,28,4,43,22,75, ipv6=,1,39,3,,,50,13,5,6,2,2,1,2,,,,, support=222,463,1419,368,343,511,443,201,442,324,291,681,757,653,174,505,687,491,
46	0.6476	0.8000	firewal=,8,98,6,4,3,72,46,9,12,13,32,5,5,,3,1,6, internet=74,184,972,140,56,257,547,183,218,280,151,267,57,84,19,102,33,89, secur=98,72,761,80,42,140,327,165,120,123,122,242,136,70,4,117,41,138,
104	0.4000	0.5000	toy=26,29,48,11,5,41,10,5,23,7,13,18,7,4,8,20,39,20, sto=170,106,196,56,24,110,86,47,63,74,24,153,1,7,6,43,12,24,
123	0.3000	0.4000	multidimension=3,62,68,43,10,20,5,12,14,18,16,13,47,114,48,86,122,28, ind=104,202,480,176,97,253,182,67,158,192,124,172,811,618,167,494,819,410, near=154,311,618,258,144,253,179,84,205,139,109,262,341,314,152,181,603,169, neighbour=1,1,,1,,1,,,,,1,,2,1,2,2,15,1, search=115,177,541,158,91,397,211,86,108,170,110,196,334,277,109,355,592,221,
176	0.3000	0.4000	secur=98,72,761,80,42,140,327,165,120,123,122,242,136,70,4,117,41,138, web=86,312,989,222,121,346,464,193,197,312,151,349,93,95,45,144,124,112, cook=19,31,64,12,3,31,32,14,8,11,12,35,12,11,25,23,7,34, authenti=,12,161,10,1,21,110,37,26,25,21,34,15,12,1,12,24,26, integr=156,375,1134,319,394,458,326,176,355,299,196,516,454,303,143,374,535,371, confidential=5,4,59,4,2,4,26,12,10,8,5,20,3,3,,10,3,19,
213	0.7000	0.8000	gib=11,2,19,11,,10,9,3,,11,1,11,2,3,12,11,93,5, sampl=50,229,350,178,122,208,71,47,149,128,56,188,207,143,138,214,685,210,

Table 5.11: CF and DF: Queries with better P@10 by local IDF strategy than global IDF in the collection organized without subject

Qid	AP_G	AP_T	Number of sub-collections containing the query terms
168	0.4029	0.4583	new=18, zealand=17, digit=18, libr=18, project=18
229	0.0504	0.0862	lat=18, semant=18, anlysi=3, lat=18, semant=18, ind=18
207	0.1836	0.2131	dom=17, sax=10
186	0.1016	0.1280	electron=18, bus=18, data=18, min=18
174	0.0398	0.0585	internet=18, web=18, pag=18, prefetch=17, algorithm=18, cpu=18, mem=18, disk=18
111	0.1706	0.1857	natur=18, languag=18, proces=18, program=18, languag=18, model=18, languag=18, human=18, languag=18
241	0.0570	0.0672	singl=18, sign=18, ldap=10
170	0.0277	0.0376	map=18, web=18
Qid	AP_G	AP_T	DF's in sub-collections
168	0.4029	0.4583	new=282,618,1775,526,474,655,503,236,534,433,338,838,947,716,206,549,951,531, zealand=12,8,34,5,4,11,7,2,2,3,1,7,,3,6,4,7,6, digit=213,326,869,173,314,223,240,102,337,326,124,168,521,205,86,186,437,119, libr=135,193,648,202,165,224,162,44,142,129,149,205,153,156,56,171,108,192, project=242,488,1184,309,212,447,300,151,310,284,213,760,292,261,160,367,624,400,
229	0.0504	0.0862	lat=260,320,946,301,267,370,268,124,340,231,158,541,496,481,100,357,457,360, semant=15,45,304,33,45,187,135,14,46,108,76,120,92,108,23,317,93,294, anlysi=,,,,1,,,,,,,,,,,1,2, lat=260,320,946,301,267,370,268,124,340,231,158,541,496,481,100,357,457,360, semant=15,45,304,33,45,187,135,14,46,108,76,120,92,108,23,317,93,294, ind=104,202,480,176,97,253,182,67,158,192,124,172,811,618,167,494,819,410,
207	0.1836	0.2131	dom=1,9,25,2,,12,29,4,4,7,3,5,5,7,4,46,26,24, sax=,,3,,1,1,5,3,,,,,7,4,1,,1,2,

186	0.1016	0.1280	electron=239,255,855,193,387,301,265,100,319,245,128,238,468,248,55,166,293,171, bus=209,178,1007,134,249,297,279,219,403,186,162,561,294,278,19,146,56,190, data=221,529,1475,420,377,558,437,213,447,373,302,657,756,641,196,585,880,486, min=90,121,312,111,68,252,70,38,113,56,102,147,392,388,87,299,408,209,
174	0.0398	0.0585	internet=74,184,972,140,56,257,547,183,218,280,151,267,57,84,19,102,33,89, web=86,312,989,222,121,346,464,193,197,312,151,349,93,95,45,144,124,112, pag=163,198,687,152,128,229,327,106,202,196,134,270,133,125,44,183,118,145, prefetch=2,4,52,3,7,2,8,,53,9,25,2,79,36,6,25,1,13, algorithm=82,380,729,392,263,401,186,43,251,181,238,188,894,727,203,487,943,385, cpu=26,82,234,93,100,33,67,13,196,43,85,39,203,149,53,127,69,98, mem=86,21,134,34,165,41,12,1,115,20,46,10,159,98,11,30,7,36, disk=81,132,303,91,40,57,70,37,123,105,88,70,101,84,37,182,81,57,
111	0.1706	0.1857	natur=196,358,671,321,157,474,197,81,187,238,159,418,468,391,143,389,689,398, languag=165,206,1046,228,200,431,288,105,220,245,205,510,281,298,43,347,184,442, proces=251,579,1626,442,456,606,416,205,534,383,333,820,926,746,204,549,972,524, program=269,437,1487,444,372,560,376,170,456,311,311,777,686,604,128,453,462,522, languag=165,206,1046,228,200,431,288,105,220,245,205,510,281,298,43,347,184,442, model=196,555,1227,441,392,557,369,143,335,325,305,644,741,663,207,518,921,514, languag=165,206,1046,228,200,431,288,105,220,245,205,510,281,298,43,347,184,442, human=151,353,605,168,45,455,189,80,103,214,81,366,45,37,104,187,481,193, languag=165,206,1046,228,200,431,288,105,220,245,205,510,281,298,43,347,184,442,
241	0.0570	0.0672	singl=154,407,939,324,306,380,310,142,391,244,237,450,793,614,166,461,756,442, sign=156,240,711,182,388,271,186,74,329,209,116,212,481,250,96,139,591,189, ldap=,1,18,3,,,34,10,,1,2,1,,1,,1,,,
170	0.0277	0.0376	map=56,408,495,175,192,265,153,67,169,189,142,218,464,422,161,322,627,277, web=86,312,989,222,121,346,464,193,197,312,151,349,93,95,45,144,124,112,

Table 5.12: CF and DF: Queries with better MAP by top QDF strategy than global IDF in the collection organized without subject

Qid	$P@10_G$	$P@10_T$	Number of sub-collections containing the query terms
93	0.0000	0.4000	charl=18, babbag=8, institut=18, inst=18
45	0.4000	0.7000	augm=18, real=18, medicin=18
42	0.4055	0.7000	decrypt=17, enigma=10, cod=18
234	0.5138	0.8000	cal=18, pap=18, confer=18, workshop=18, multimedia=18
46	0.6476	0.9000	firewal=16, internet=18, secur=18
110	0.6000	0.8000	stream=18, deliv=18, stream=18, synchroniz=18, audio=18, video=18, stream=18, appli=18
111	0.5000	0.7000	natur=18, languag=18, proces=18, program=18, languag=18, model=18, languag=18, human=18, languag=18
123	0.3000	0.5000	multidimension=18, ind=18, near=18, neighbour=11, search=18
176	0.3000	0.5000	secur=18, web=18, cook=18, authenti=17, integr=18, confidential=17
186	0.1000	0.3000	electron=18, bus=18, data=18, min=18
206	0.4000	0.6000	problem=18, phys=18, limit=18, miniaturiz=16, microproces=18
213	0.7000	0.9000	gib=16, sampl=18
216	0.5000	0.7000	multimedia=18, retriev=18, system=18, architectur=18
33	0.6201	0.8000	softwar=18, pat=18
39	0.6328	0.8000	video=18, demand=18
50	0.6757	0.8000	xml=18, edit=18, pars=18
102	0.3000	0.4000	distribut=18, storag=18, system=18, grid=18, comput=18
163	0.8000	0.9000	multi=18, agen=18, network=18, internet=18
165	0.2000	0.3000	techno=18, disabl=18, handicap=17, peopl=18
174	0.1000	0.2000	internet=18, web=18, pag=18, prefetch=17, algorithm=18, cpu=18, mem=18, disk=18
207	0.7000	0.8000	dom=17, sax=10
212	0.8000	0.9000	hmm=13, hidden=18, markov=17, model=18, equ=18
218	0.5000	0.6000	comput=18, assist=18, compos=18, mus=18, not=18, midi=10
229	0.2000	0.3000	lat=18, semant=18, anlysi=3, lat=18, semant=18, ind=18
43	0.3000	0.4000	approxim=18, str=18, match=18, algorithm=18
Qid	$P@10_G$	$P@10_T$	DF's in sub-collections
43	0.3000	0.4000	approxim=108,271,332,289,130,250,81,32,169,96,126,172,493,418,180,290,822,253, str=27,36,169,46,19,88,62,15,28,41,30,46,104,82,13,128,140,107, match=45,228,442,127,112,249,141,50,176,118,93,193,306,307,80,276,644,244,

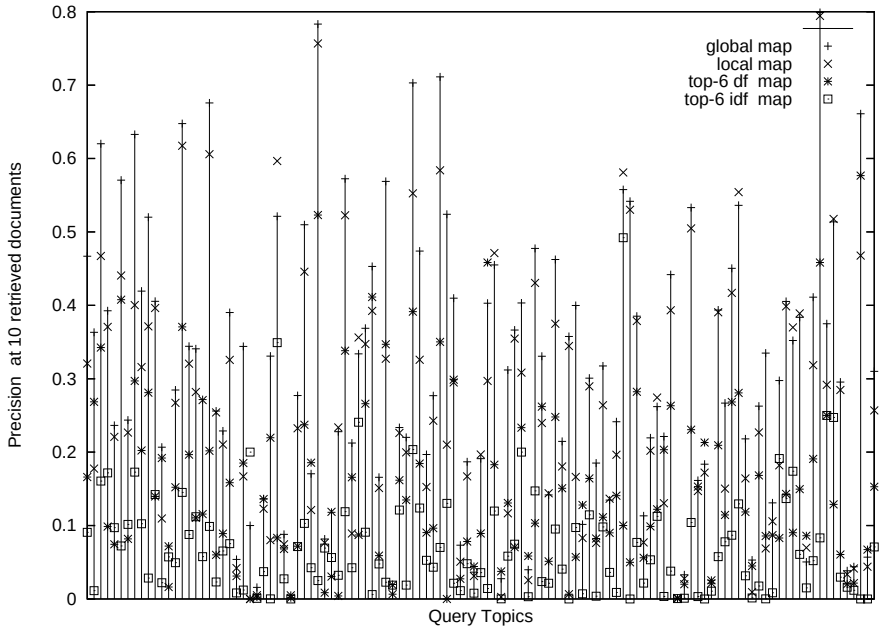
			algorithm=82,380,729,392,263,401,186,43,251,181,238,188,894,727,203,487,943,385,
229	0.2000	0.3000	lat=260,320,946,301,267,370,268,124,340,231,158,541,496,481,100,357,457,360, semant=15,45,304,33,45,187,135,14,46,108,76,120,92,108,23,317,93,294, anlysi=,,,,1,,,,,,,,,,,1,2, lat=260,320,946,301,267,370,268,124,340,231,158,541,496,481,100,357,457,360, semant=15,45,304,33,45,187,135,14,46,108,76,120,92,108,23,317,93,294, ind=104,202,480,176,97,253,182,67,158,192,124,172,811,618,167,494,819,410,
218	0.5000	0.6000	comput=316,680,1902,571,539,677,547,204,554,417,358,777,1042,756,225,571,1030,553, assist=82,109,216,61,48,157,60,28,54,59,34,98,82,41,59,89,77,111, compos=53,155,273,83,73,131,105,19,80,121,85,115,287,211,84,223,326,284, mus=42,67,104,21,10,51,44,12,32,111,14,41,5,3,5,19,25,21, not=301,597,1756,545,462,646,507,241,542,387,308,909,1000,745,202,553,998,538, midi=,7,10,3,1,4,,,2,22,,3,1,,,,1,,
212	0.8000	0.9000	hmm=1,3,7,1,,6,2,,3,3,1,4,,,,4,67,1, hidden=27,76,138,47,14,69,33,14,47,40,31,66,62,64,41,78,191,79, markov=3,6,24,25,6,31,4,,10,7,2,9,99,43,3,33,233,42, model=196,555,1227,441,392,557,369,143,335,325,305,644,741,663,207,518,921,514, equ=142,309,446,371,206,218,121,59,226,119,133,256,855,657,182,444,883,389,
207	0.7000	0.8000	dom=1,9,25,2,,12,29,4,4,7,3,5,5,7,4,46,26,24, sax=,,3,,1,1,5,3,,,,,7,4,1,,1,2,
174	0.1000	0.2000	internet=74,184,972,140,56,257,547,183,218,280,151,267,57,84,19,102,33,89, web=86,312,989,222,121,346,464,193,197,312,151,349,93,95,45,144,124,112, pag=163,198,687,152,128,229,327,106,202,196,134,270,133,125,44,183,118,145, prefetch=2,4,52,3,7,2,8,,53,9,25,2,79,36,6,25,1,13, algorithm=82,380,729,392,263,401,186,43,251,181,238,188,894,727,203,487,943,385, cpu=26,82,234,93,100,33,67,13,196,43,85,39,203,149,53,127,69,98, mem=86,21,134,34,165,41,12,1,115,20,46,10,159,98,11,30,7,36, disk=81,132,303,91,40,57,70,37,123,105,88,70,101,84,37,182,81,57,
165	0.2000	0.3000	techno=257,491,1564,360,447,582,439,215,495,380,310,665,641,532,134,387,531,388, disabl=6,18,53,4,27,23,21,5,44,14,13,16,52,34,8,17,13,50, handicap=9,5,6,6,1,10,2,,8,6,1,4,8,3,1,5,5,1, peopl=230,270,890,206,111,372,260,167,217,235,101,625,54,33,41,122,146,172,
163	0.8000	0.9000	multi=12,64,98,41,38,92,103,10,52,45,36,41,191,175,56,145,258,89, agen=111,121,566,107,47,399,265,75,89,132,118,197,80,89,17,174,110,177, network=140,309,1236,256,187,483,453,184,353,348,298,367,650,664,66,364,422,323, internet=74,184,972,140,56,257,547,183,218,280,151,267,57,84,19,102,33,89,
102	0.3000	0.4000	distribut=92,261,916,264,156,331,365,118,208,245,337,328,641,765,105,403,382,389, storag=126,145,465,120,89,120,121,60,186,154,126,99,273,206,70,254,141,133, system=266,606,1780,494,504,696,508,234,559,436,357,875,970,765,198,565,935,548, grid=18,161,117,163,28,48,22,5,44,24,49,24,73,165,131,45,229,29, comput=316,680,1902,571,539,677,547,204,554,417,358,777,1042,756,225,571,1030,553,
50	0.6757	0.8000	xml=1,13,146,10,7,57,149,53,9,37,9,19,1,2,2,16,2,9, edit=254,287,781,320,215,320,244,79,199,248,159,427,329,317,91,262,310,310, pars=7,24,123,19,12,84,81,17,17,38,26,49,30,27,8,64,38,87,
39	0.6328	0.8000	video=61,303,497,60,70,126,143,54,168,372,75,87,80,62,44,105,248,41, demand=124,164,597,157,142,210,182,122,218,178,131,304,198,191,48,140,123,151,
33	0.6201	0.8000	softwar=197,467,1603,368,305,495,417,207,442,319,310,936,460,474,122,359,201,570, pat=92,60,222,23,67,45,49,13,118,35,13,62,160,69,30,55,82,31,
216	0.5000	0.7000	multimedia=21,201,530,32,71,108,148,41,152,465,106,87,123,127,40,225,90,69, retriev=51,97,351,67,15,245,154,37,58,177,60,93,84,87,41,325,197,122, system=266,606,1780,494,504,696,508,234,559,436,357,875,970,765,198,565,935,548, architectur=124,248,1081,217,318,357,352,115,382,222,284,404,716,632,76,311,200,325,
213	0.7000	0.9000	gib=11,2,19,11,,10,9,3,,11,1,11,2,3,12,11,93,5, sampl=50,229,350,178,122,208,71,47,149,128,56,188,207,143,138,214,685,210,
206	0.4000	0.6000	problem=250,454,1400,492,384,576,394,205,400,293,290,774,842,692,195,522,965,515, phys=118,243,318,344,83,167,75,18,147,81,87,121,169,175,111,103,316,102, limit=186,397,1031,320,291,417,310,130,351,248,229,481,662,521,160,418,757,427, miniaturiz=12,16,33,11,9,9,4,3,19,4,4,,6,1,,1,1,1, microproces=46,30,347,38,204,22,19,8,268,19,54,33,218,97,5,9,6,28,
186	0.1000	0.3000	electron=239,255,855,193,387,301,265,100,319,245,128,238,468,248,55,166,293,171, bus=209,178,1007,134,249,297,279,219,403,186,162,561,294,278,19,146,56,190, data=221,529,1475,420,377,558,437,213,447,373,302,657,756,641,196,585,880,486,

			min=90,121,312,111,68,252,70,38,113,56,102,147,392,388,87,299,408,209,
176	0.3000	0.5000	secur=98,72,761,80,42,140,327,165,120,123,122,242,136,70,4,117,41,138, web=86,312,989,222,121,346,464,193,197,312,151,349,93,95,45,144,124,112, cook=19,31,64,12,3,31,32,14,8,11,12,35,12,11,25,23,7,34, authenti=,12,161,10,1,21,110,37,26,25,21,34,15,12,1,12,24,26, integr=156,375,1134,319,394,458,326,176,355,299,196,516,454,303,143,374,535,371, confidential=5,4,59,4,2,4,26,12,10,8,5,20,3,3,,10,3,19,
123	0.3000	0.5000	multidimension=3,62,68,43,10,20,5,12,14,18,16,13,47,114,48,86,122,28, ind=104,202,480,176,97,253,182,67,158,192,124,172,811,618,167,494,819,410, near=154,311,618,258,144,253,179,84,205,139,109,262,341,314,152,181,603,169, neighbour=1,1,,1,,1,,,,,1,,2,1,2,2,15,1, search=115,177,541,158,91,397,211,86,108,170,110,196,334,277,109,355,592,221,
111	0.5000	0.7000	natur=196,358,671,321,157,474,197,81,187,238,159,418,468,391,143,389,689,398, languag=165,206,1046,228,200,431,288,105,220,245,205,510,281,298,43,347,184,442, proces=251,579,1626,442,456,606,416,205,534,383,333,820,926,746,204,549,972,524, program=269,437,1487,444,372,560,376,170,456,311,311,777,686,604,128,453,462,522, languag=165,206,1046,228,200,431,288,105,220,245,205,510,281,298,43,347,184,442, model=196,555,1227,441,392,557,369,143,335,325,305,644,741,663,207,518,921,514, languag=165,206,1046,228,200,431,288,105,220,245,205,510,281,298,43,347,184,442, human=151,353,605,168,45,455,189,80,103,214,81,366,45,37,104,187,481,193, languag=165,206,1046,228,200,431,288,105,220,245,205,510,281,298,43,347,184,442,
110	0.6000	0.8000	stream=32,120,315,58,56,85,128,39,130,190,86,55,182,115,37,81,115,81, deliv=108,121,511,81,108,140,201,107,150,168,99,308,152,215,15,75,33,155, stream=32,120,315,58,56,85,128,39,130,190,86,55,182,115,37,81,115,81, synchroniz=16,77,248,58,71,42,96,25,114,139,141,64,233,330,18,93,22,175, audio=45,137,265,29,32,59,96,27,98,275,41,31,34,15,7,55,37,30, video=61,303,497,60,70,126,143,54,168,372,75,87,80,62,44,105,248,41, stream=32,120,315,58,56,85,128,39,130,190,86,55,182,115,37,81,115,81, appli=227,680,1625,499,444,615,474,212,503,419,343,741,939,713,207,550,973,529,
46	0.6476	0.9000	firewal=,8,98,6,4,3,72,46,9,12,13,32,5,5,,3,1,6, internet=74,184,972,140,56,257,547,183,218,280,151,267,57,84,19,102,33,89, secur=98,72,761,80,42,140,327,165,120,123,122,242,136,70,4,117,41,138,
234	0.5138	0.8000	cal=237,455,1227,373,308,500,390,162,404,321,263,578,854,688,182,513,809,482, pap=237,229,535,162,195,281,181,54,171,172,118,286,987,750,218,559,971,543, confer=158,142,474,101,245,213,137,51,96,154,95,234,316,363,70,274,243,250, workshop=42,118,386,93,210,248,121,22,83,136,116,151,324,337,113,316,444,336, multimedia=21,201,530,32,71,108,148,41,152,465,106,87,123,127,40,225,90,69,
42	0.4055	0.7000	decrypt=15,6,74,2,7,5,26,10,15,12,4,8,16,4,,3,1,13, enigma=26,2,2,1,,6,1,,3,,3,2,,,,,,1, cod=154,256,1090,339,199,276,283,128,321,213,209,638,493,338,97,240,402,389,
45	0.4000	0.7000	augm=24,94,157,39,36,101,43,11,50,65,36,52,125,110,39,128,113,103, real=210,574,1307,390,274,534,363,183,406,367,267,695,549,451,178,426,798,442, medicin=35,84,106,67,2,94,15,9,10,33,16,35,7,10,37,49,81,18,
93	0.0000	0.4000	charl=166,31,178,47,34,51,59,16,19,26,25,54,31,50,10,12,30,15, babbag=136,,14,,2,1,1,,4,,,5,,,,1,,, institut=217,281,633,243,193,373,173,63,192,168,124,371,475,444,141,295,526,304, inst=34,38,147,88,12,154,18,9,22,34,63,106,123,109,47,82,180,103,

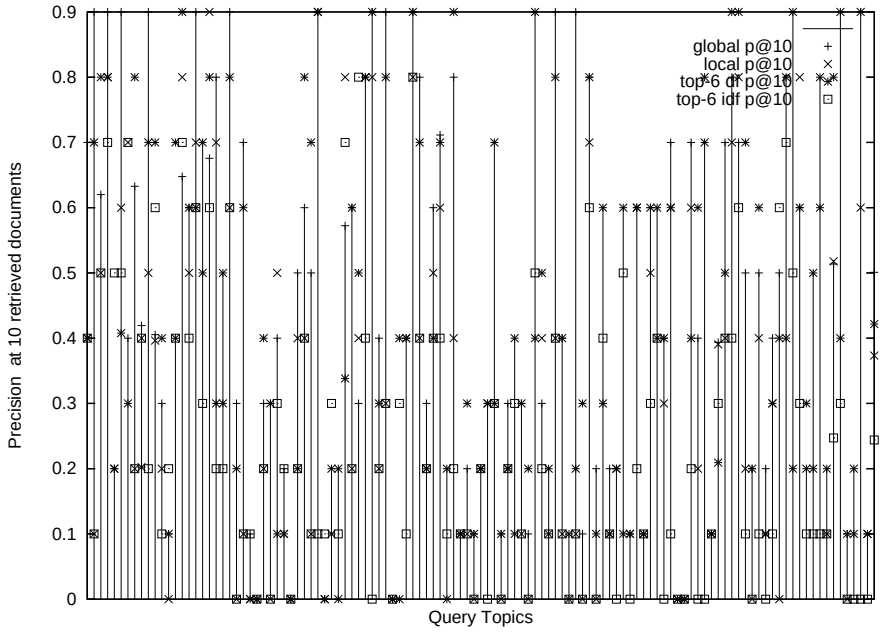
Table 5.13: CF and DF: Queries with better P@10 by top QDF strategy than global IDF in the collection organized without subject

5.7.6 Element Retrieval with TF-IDF

Although in the definition of context-specific IDF, IDF can be computed based on any non-leaf node of the the collection tree, due to the computational cost we only define three types of IDF: inverse document frequency, inverse section frequency and inverse paragraph frequency to retrieve three types of elements: document, section and paragraph. The smallest element in our



(a) AP



(b) P@10

Figure 5.5: Comparison of single query performance results from global IDF, local IDF, top QDF and top QInf strategies in the multi-collection organized without subject

experiment is the paragraph.

For the retrieval procedure:

(1)match all three types of elements containing query terms

(2)compute the RSV's for all the elements, the rule for IDF choosing is as the following:

if the element is a paragraph, then use $idf_{para}(t, c_{journal-year})$

if the element is a section, then use $idf_{sec}(t, c_{journal})$

if the element is a document, then use $idf_{doc}(t, c_{INEX})$

(3)re-rank the elements according to their RSV's

To compare context-specific TF-IDF performance to the classical TF-IDF with the single type IDF, we set up the runs as follows:

(1) Global IDF: using IDF computed based on the whole collection.

(2) Local IDF: using IDF computed based on the journal sub-collection to which the element belongs.

(3) Mixed IDF: using global IDF for document, local journal-specific IDF for section, and local journal-year-specific IDF for paragraph.

IDF's used in global and local strategies are all computed based three element type separately: document, section and paragraph.

For the TF part, we still use maximum, element length, Poisson based normalization. Here maximum normalization is $tf_{max}(t,e) = \frac{n_L(t,e)}{max_{t' \in e}(n_L(t',e))}$, length normalization is $tf_{sum}(t,e) = \frac{n_L(t,e)}{N_L(e)}$, and Poisson normalization is the same to document retrieval $tf_{Poissona} = \frac{n_L(t,e)}{n_L(t,e)+1}$.

Element retrieval results in table 5.14 show that retrieval strategy with Poisson approximated normalized TF and document based global IDF has the best MAP. Poisson based TF has better performance in retrieval strategies comparing to other TF's. Element length normalized TF has extremely low MAP and P@10, the reason may be that the element length normalized TF is biased to small elements, and ranks the small elements too high. In fact the small element may not be judged as relevant during the assessment due to it lacking enough information, while their parent elements are more appropriate to be judged as relevant.

retrieval function	global idf(t,INEX)		local idf(t,journal)		mix idf(t,journal)	
	MAP	P@10	MAP	P@10	MAP	P@10
idf_{doc}, tf_{max}	0.0279	0.3284	0.0150	0.2078	0.0191	0.5112
idf_{doc}, tf_{sum}	0.0063	0.1052	0.0051	0.0802	0.0164	0.4078
idf_{doc}, $tf_{Poissona}$	***0.0533***	0.5888	0.0281	0.3500	0.0208	0.5793
idf_{sec},tf_{max}	0.0261	0.3284	0.0162	0.2560	0.0229	0.3509
idf_{sec},tf_{sum}	0.0036	0.0948	0.0030	0.0621	0.0143	0.1784
idf_{sec} $tf_{poissona}$	0.0470	0.5931	0.0300	0.4060	0.0293	0.4534
idf_{para},tf_{max}	0.0244	0.3241	0.0159	0.2638	0.0245	0.4104
idf_{para},tf_{sum}	0.0028	0.0828	0.0024	0.0534	0.0114	0.1198
idf_{para} $tf_{poissona}$	0.0441	***0.5957***	0.0293	0.4060	0.0324	0.4845
idf_{loc},tf_{max}	0.0225	0.3216	0.0180	0.2862	0.0258	0.4017
idf_{loc},tf_{sum}	0.0024	0.0836	0.0022	0.0603	0.0039	0.0853
idf_{loc} $tf_{poissona}$	0.0406	0.5922	0.0333	0.4793	0.0386	0.5052
$idf_{d,s,p}$,tf_{max}					0.0258	0.4500
$idf_{d,s,p}$,tf_{sum}					0.0112	0.1129
$idf_{d,s,p}$,$tf_{Poissona}$					0.0317	0.5862

Table 5.14: TF-IDF element retrieval with context-specific IDF

Context-specific IDF in element retrieval, which chooses type-specific and context-specific IDF according to the type of the retrieved element, does not have good MAP, whereas the P@10 value is similar to the best P@10 value.

When we look into each query's ranking list, we find that all the runs tend to rank documents in high ranks. Because when the document contain more query terms than the paragraph, $tf(t,d)$ is greater than $tf(t,e)$, as a result the document's RSV is always greater than paragraph's. This is the case for global and local IDF strategy. Mixed IDF strategy will give some lift on the RSV's of those elements that contain the rare terms in term of section frequency or paragraph frequency. However TF normalization seems to play more important role in element retrieval with TF-IDF function.

Because the element retrievals rank the document higher than element, which makes top 10 the retrieval result list is similar to document retrieval, and have similar average precisions as document retrieval. While element retrievals return too long ranking lists, so that the MAPs are very low.

For context-specific IDF, there are two aspects that determined that our hypothesis did not come true. The first is that pure TF-IDF retrieval functions regardless IDF type tends to rank documents in high rank as we discussed in the previous paragraph. The second is due to the system limit, that we only maintain the document, section and paragraph frequency spaces with respect to INEX collection, journal sub-collection and journal+year sub-collection. IDF on the sub-

collection can show the term's discriminativeness for section or paragraph in the sub-collection , but not in the document. This result suggests that combining the IDF within the collection and IDF within the document could better rank the elements and documents. However it will in turn unavoidably incur some parameters for the combination, which is deviating from our original target parameter-free.

Next, lets look at context-specific frequencies in LM. As for LM, the probability that a document generate a query term is based only on location frequency, therefore, there are much less runs than what we have with TF-IDF. We show both LM document retrieval and element retrieval result in the same section.

5.7.7 Context-specific Frequencies in LM

Using LM in structured retrieval has been explored by [Si et al., 2002], [Kamps et al., 2004], and [Ogilvie and Callan, 2003] etc. [Si et al., 2002] used LM to weight sub-collection, and smooth the document weight by the collection weight, [Kamps et al., 2004] applied LM to XML retrieval, whose retrieval objects varied from document to element. The important impact of this work is the introducing element length prior into retrieval model.

We just use a unified LM framework to retrieve both document and element. Global or local frequencies, based either document or element, can be used for smoothing. When simulating the distributed retrieval, we use the same strategies as what used in context-specific TF-IDF retrieval: QDF and QInf

As parameter tuning is not the main focus of our work, we set the parameter λ to classical value 0.2, which has general good performance across the collections (see [Hiemstra, 2001]).

Same to the experiment in TF-IDF, the document collection is either grouped by Journal (subject), or year (without subject).

The smoothing strategies we use are:

(1) Global DF: Using DF, EF and LF to smooth document model.

(2) Local DF: Using DF, EF and LF within a sub-collection to smooth document model

(3) Top promising collections: choose promising collections, then use their local DF, EF, LF to smooth document model.

LM	global *df*(t,INEX)		local *df*(t,journal)		Top-6 QDF(q,Journal)		Top-6 QInf(q,Journal)	
smoothing with	MAP	P@10	MAP	P@10	MAP	P@10	MAP	P@10
DF	0.2277	0.4078	0.1724	0.2914	0.1749	0.3784	0.0229	0.1078
SF	0.2418	0.4138	0.1510	0.2586	0.1563	0.3207	0.0184	0.0983
PF	0.2556	0.4414	0.1388	0.2328	0.1613	0.3560	0.0133	0.0776
LF	***0.3210***	***0.5388***	0.1999	0.3750	0.1934	0.4243	0.0237	0.1035

Table 5.15: LM document retrieval with context-specific (by journal, with subject) frequencies smoothing

LM	global *df*(t,INEX)		local *df*(t,year)		Top-3 QDF(q,year)		Top-3 QInf(q,year)	
smoothing with	MAP	P@10	MAP	P@10	MAP	P@10	MAP	P@10
DF	0.2277	0.4078	0.2201	0.3931	0.1074	0.3112	0.0808	0.2698
SF	0.2418	0.4138	0.2303	0.3871	0.1105	0.3095	0.0816	0.2733
PF	0.2556	0.4414	0.2365	0.4060	0.1235	0.3371	0.0760	0.2716
LF	***0.3210***	***0.5388***	0.3049	0.5216	0.1603	0.4360	0.0951	0.3096

Table 5.16: LM document retrieval with context-specific (by year, without subject) frequencies smoothing

LM document retrieval results in table 5.15 and 5.16 show that $P(t|d)$ smoothing with LF can produce the best result. With the granularity diminishing, smoothing with document, section, paragraph and location based $P(t|c)$ respectively increases the retrieval precision. This is different from conclusion in [Hiemstra, 2001], which claimed that document model smoothing with document based $P(t|c)$ outperforms the run smoothing with location based $P(t|c)$.

LM smoothing with respect to global, local and top QDF promising collection strategy, shows the same behavior to TF-IDF model that global strategy is the best, then local strategy, and then top QDF promising strategy. Smoothing with global DF, EF and LF is better than smoothing with local DF, EF and LF respectively. When the collection is organized by subject, the local strategy works similarly to top QDF promising strategy, but all are worse than global strategy. While there is no subject in each sub-collection, the local strategy works similarly to global strategy, and better than top QDF promising strategy.

Next we show the element retrieval results in table 5.17:

LM-element	global *df*(t,INEX)		local *df*(t,journal)		local *df*(t,year)		mix *df*	
smoothing with	MAP	P@10	MAP	P@10	MAP	P@10	MAP	P@10
DF	0.0126	0.1664	0.0070	0.1112	0.0118	0.1543	0.0070	0.1211
SF	0.0152	0.2043	0.0076	0.1241	0.0138	0.1871	0.0073	0.1155
PF	0.0195	0.2759	0.0083	0.1578	0.0171	0.2638	0.0074	0.1276
LF	**0.0358**	**0.4043**	0.0199	0.3276	0.0330	0.3983	0.0195	0.3155
mixed DF							0.0053	0.1276

Table 5.17: LM element retrieval with context-specific DF

In LM element retrieval, still the retrieval strategy which smooths the element model $P(t|e)$ with global frequencies has a better retrieval result, and location based smoothing works the best. Similar to LM document retrieval, local strategy works better on the collection organized without subject than with subject. The reason should be the same as has been discussed before.

5.8 Variance of LF as A New Discriminativeness Measurement

IDF indicates the discriminativeness of term, so does ILF to a certain degree. Consideration of the average allows for measuring the randomness of a term as the deviation (variance) of the actual document location frequency from the expected document location frequency $lf(t,c)$ ([Amati and van Rijsbergen, 2002]).

$$\begin{aligned} lf(t,c) &:= \frac{n_L(t,c)}{N_L(c)} \\ lf(t,d) &:= \frac{n_L(t,d)}{N_L(d)} \\ \sigma_L^2(t,c) &:= \frac{1}{N_D(c)} \sum_d (lf(t,d) - lf(t,c))^2 \end{aligned} \tag{5.14}$$

The deviation $\sigma_L^2(t,c)$ (we use here the old and intuitive $1/N_D(c)$ factor in the deviation but $1/(N_D(c)-1)$ would be the statistically correct factor; however, this is of minor importance for the discussion here.) is small for randomly distributed terms, and large for the other terms. One of the fundamental observations of term statistics is that function words (stop-words, terms that are not discriminative, and do not lead to a high chance for relevant documents) are randomly distributed, whereas "good" terms (terms that lead to relevant documents) are not randomly distributed ([Church and Gale, 1995], [Amati and van Rijsbergen, 2002], others).

Formulating this observation in a mathematical theorem, the claim is that the randomness measure and the discriminativeness measure are correlated:

$$\sigma_L^2(t_1,c) > \sigma_L^2(t_2,c) \iff idf(t_1,c) > idf(t_2,c) \tag{5.15}$$

As in structured document retrieval elements can be viewed as the intermediate between locations and documents, we can formulate the randomness measure with respect to elements:

$$\begin{aligned} ef(t,c) &:= \frac{n_E(t,c)}{N_E(c)} \\ ef(t,d) &:= \frac{n_E(t,d)}{N_E(d)} \\ \sigma_E^2(t,c) &:= \frac{1}{N_D(c)} \sum_d (ef(t,d) - ef(t,c))^2 \end{aligned} \tag{5.16}$$

Definition 5.16 for the randomness measure with respect to the element frequency is analogous to definition 5.14 for the randomness with respect to the location frequency.

Analogously to rule 5.15, we formulate a rule for location randomness $\sigma_L^2(t,c)$ and $ief(t,c)$, and a rule for element randomness $\sigma_E^2(t,c)$ and $idf(t,c)$.

$$\sigma_L^2(t_1,c) > \sigma_L^2(t_2,c) \iff ief(t_1,c) > ief(t_2,c) \tag{5.17}$$

$$\sigma_E^2(t_1,c) > \sigma_E^2(t_2,c) \iff idf(t_1,c) > idf(t_2,c) \tag{5.18}$$

The investigation result of theorem 5.15 based on INEX collection is as the follows:

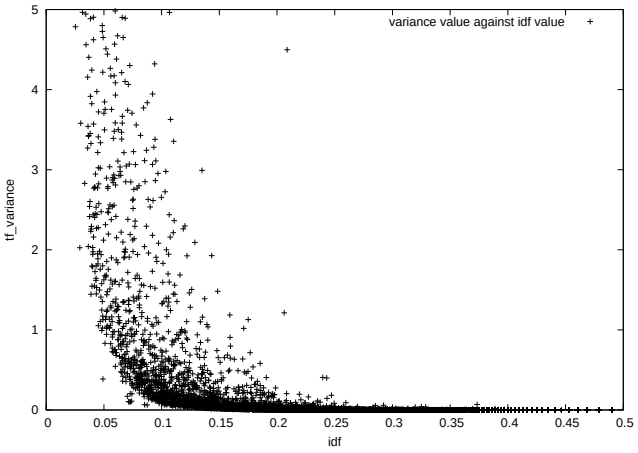


Figure 5.6: IDF value against TF variance value

There is no correlation between IDF and TF variance, but a highly negative correlation between their ranks. This result does not confirm our assumption about IDF and TF variance, which is also good as it may be used into improve the performance of IDF.

5.9 Summary

We have presented and investigated a new generalized retrieval model for structured document retrieval in this chapter. The basic idea of the model is context-specific frequencies and discriminativeness. By context-specific frequencies we mean that the retrieval function choose appro-

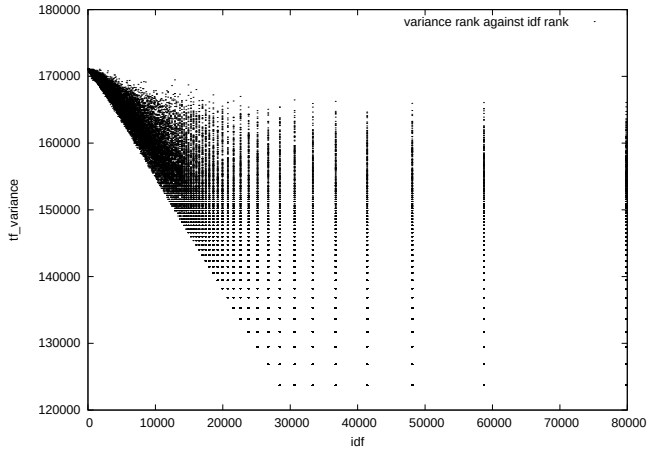


Figure 5.7: IDF rank against TF variance rank

priate size of surrounding text for the retrieval object to count the term, element or document frequencies, also the type of frequency to compute the term discriminativeness or to smooth term weights. This is different to classical retrieval functions which usually use global frequencies.

The motivation for the context-specific discriminativeness comes from the observation that in a structured collection, a term might be relatively frequent in one sub-collection, whereas the same term might be relatively rare in another sub-collection. Then, the term should have a relatively small effect on the RSV of an element retrieved from the sub-tree in which the term is frequent, and, on the other hand, the term should have a relatively strong effect on the RSV of an element retrieved from the sub-tree in which the term is rare.

The main experimental finding is:

> The retrieval function based on context-specific discriminativeness is a generalization of classical retrieval model, and does not require any of the heuristic parameters for term weight or RSV propagation as those alternative approaches in structured document retrieval.
>
> For document retrieval in the collection organized by subject, global IDF or LM smoothed with global frequency have better performance than local IDF or LM smoothed with local frequency. However if the collection is not organized by subject, each sub-collection contains various subjects, then global IDF has similar performance to local IDF. Similar results of LM are obtained in the experiments.
>
> For TF-IDF, the document based IDF performs the best; whilst For LM, $P(t|d)$ smoothed with document based probability $P(t|c)$ performs the worst compared to

element and location based probability $P(t|c)$, smoothing with location based $P(t|c)$ works the best.

Whether a collection is organized by subject has a strong impact on local or top QDF promising strategy. If the collection is organized by subject, then local strategy works similarly to top QDF promising strategy, but worse than global strategy. If there is no subject in each sub-collection, then local strategy works similar to global strategy, and much better than top QDF promising strategy.

To select the most promising sub-collections, the number of documents containing query terms in the sub-collection is a good criteria. The sub-collection that has more documents containing query terms tends to have more relevant document. The sub-collections chosen by query informativeness are less likely to contain relevant documents.

When doing context-specific IDF document retrieval experiments, we also tried different TF normalization: maximum TF normalization, length normalization and Poisson approximate normalization. Different combinations of IDF and TF yield different results. Poisson normalized TF and global IDF always has the best performance.

The experiment in this chapter confirms that global IDF yields the best retrieval quality despite the intuition that IEF or context-specific frequencies cover better the specialties of element retrieval.

In current experimental setting, we maintain a discriminativeness space for each sub-collection. In theory, we would like to maintain a discriminativeness space per node in a structured document collection, which requires significant resources and has not been implemented yet. We find that with our experiment setting, purely relying on discriminativeness from a single context would not improve the retrieval performance, different context-specific frequencies should be combined in order to improve the quality of element retrieval. The next research steps could include the development and investigation of dynamical combination of context-specific frequencies.

Chapter 6

Model selection Based on Correlation of Query Statistical Features and Performance

---

TF-IDF and Language Modelling (LM) are retrieval models with stable and good performance over sets of queries. For some queries, TF-IDF performs better, for others, LM is superior. Therefore, one idea to improve the overall performance of a retrieval system is to choose for each query the model that is likely to perform best. In this chapter, we investigate the ranking correlation of TF-IDF and LM, as well as the correlation of statistical query features (SQF) and query performance, in order to identify the query features that make TF-IDF better than LM, or vice versa. We focus on the average term frequency (AvgTF) and its related statistical query features: number of documents with TF greater than AvgTF, percentage of documents with TF greater than AvgTF, and relevant entropy of TF distribution and Poisson distribution.

This chapter is structured as follows. In section 6.1 we introduce the motivation for ranking correlation study. In section 6.2 we present the background on research of query performance prediction, and correlation test methods used in this thesis. Studies on average TF and TF distribution are shown in section 6.3. The mathematical analysis of the retrieval models, and the reason using correlation test is described in section 6.4. In section 6.5 we show the model selection experiments based on SQF, the empirical analysis of ranking correlation of retrieval models, the statistical query feature values of selected queries, and the analysis the experiment results. The summary and future work are discussed in section 6.6.

6.1 Motivation

To improve retrieval performance, IR researchers continually develop new retrieval models and study parameter estimation in retrieval models. However, this can only improve the retrieval qualities over some queries, not all queries. Even if a model has good performance in one collection, it is not guaranteed to have the same good performance in another collection. The main reason is that some models have a bias to certain kinds of query terms. Hence, if we choose an appropriate retrieval model for each query based on its statistical query features against the data collection, then the overall performance can be improved without further elaborating the models.

Previous studies show statistical query features (SQF) are correlated to the performance of retrieval models. [He and Ounis, 2004] used statistical query features to cluster the queries, and showed that queries in the same cluster perform the same, which indicates that they are favoring the same retrieval models. [Cronen-Townsend et al., 2002] used clarity score to identify difficult queries, which is more correlated to average precision than IDF (the correlation coefficients are from 0.368 to 0.577 in TREC2-8). The query's clarity score is Kullback-Leibler distance (or relevant entropy) between the term distribution in query and collection. [Amati et al., 2004] applied an information function as a query difficulty prediction in query expansion, and obtained better performance by selective query expansion than query expansion for all the queries.

In this study, we choose TF-IDF and LM as the candidates to do the model selection, because the two models have simple forms and robust performances. It is also due to the fact that TF-IDF and LM have different retrieval qualities for the same query on the same collection, which is necessary for our study. The candidate models are not restricted only to TF-IDF and LM, as long as the two models work differently, we can choose a performing retrieval model for a query. As a result we can achieve better average precision with mixing two models than with any single model. The question is which SQF should be used in the model selection.

To address the problem, we focus on average term frequency (AvgTF) and within document term frequency (TF) distribution. AvgTF of term t is the average term frequency among the documents containing term t, denoted by $avgtf(t,c) := \frac{n_L(t,c)}{n_D(t,c)}$, where $n_L(t,c)$ denotes the locations where term t occurs in the collection, and $n_D(t,c)$ stands for number of documents containing term t. Usually the discriminativeness of a term is decided by its inverse document frequency (IDF) within the collection, as a less frequently occurring term is good to discriminate documents. IDF has been proved as an effective discriminativeness measurement. However, we

wonder whether two terms t_1 and t_2 are equally discriminative if they occur in the same amount of documents, but with different distributions? For example t_1 occurs in $n_D(t_1,c) = 10,000$ documents with $avgtf(t_1,c) = 2$, t_2 occurs in $n_D(t_2,c) = 10,000$ documents with $avgtf(t_2,c) = 4.5$. Further, what if term t_1 and t_2 have the same AvgTF 2, but t_1 is evenly distributed in these 10,000 documents, while t_2 is mainly concentrated in 100 documents and occurs only once in each of the remaining 9,900? The intuitive answer to the question is that AvgTF and TF distribution could have an impact on the performances of retrieval models. Salton, Church, and Kwok's work also support the idea: [Salton et al., 1975] showed that a good term is neither a very rare nor very frequent term; [Church and Gale, 1995] showed that good terms are far from having a Poisson distribution, have more documents with high TF than expected by a Poisson distribution; [Kwok, 1996] showed that average TF and peaked IDF can improve MAP; [Kwok, 2005] showed that combined IDF and the distribution of average TF can better predict the query difficulty, especially for short queries.

In this study, we propose a method using the statistical query features to choose a suitable retrieval model for each query. With two retrieval models TF-IDF and LM, we aim to divide the training queries set into 3 groups. In one group, TF-IDF performs better than LM (i.e. average precision of TF-IDF is greater than average precision of LM); in the other group, LM performs better (i.e. average precision of LM is greater than average precision of TF-IDF); and in the last group, TF-IDF and LM perform similarly. Then we will check which Statistic query feature can better divide the queries into groups, and what is threshold for the grouping.

Based on the assumption there is a correlation between a SQF and retrieval quality, we expect that the ideal relationship between the SQF and retrieval quality - average precision (AP) - should be as shown in figure 6.1: when the query's SQF is less than a threshold value v_2, LM performs better on this query; when the query's SQF is higher than v_1, TF-IDF performs better; and for query having SQF between v_1 and v_2, either TF-IDF or LM can perform better on this query. Note the threshold v_1 and v_2 are collection dependent, can be obtained empirically. By running the TF-IDF and LM separately on a training query set, and evaluating the retrieval results, we can group the queries into three classes according to average precision. Subsequently, the threshold of query features can be identified.

With the identified thresholds, the queries with SQF greater than v_1 can be processed with TF-IDF, the queries with SQF less than v_2 can be processed with LM, and the queries having SQF

between v_1 and v_2 can be retrieved by any retrieval model, but only one model should be applied. As the queries with SQF between v_1 and v_2 can be applied to any retrieval model, therefore, these queries can be merged into either group with SQF greater than v_1 or less less than v_2. As a result, only one threshold is required to choose a proper retrieval model for the query. This setting will hopefully lead to better retrieval performance, as each query is assigned a retrieval model which works best for it.

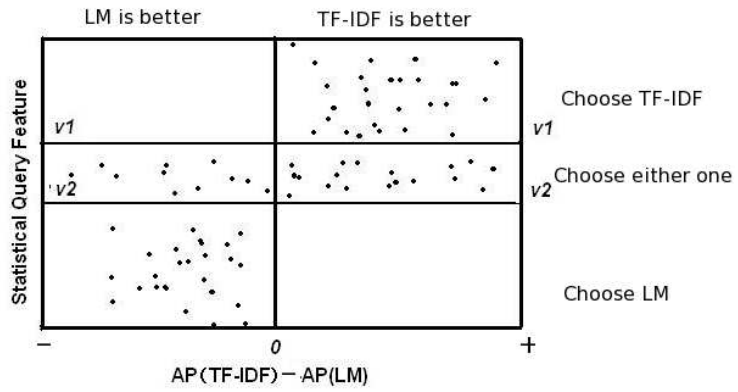


Figure 6.1: The ideal scenario in which the queries can be grouped into 3 classes according to SQF and AP: When SQF is greater than v_1, TF-IDF performs better than LM; When SQF is less than v_2, LM performs better than TF-IDF; When SQF is between v_1 and v_2, there is no conclusion whether TF-IDF or LM has a better performance, any retrieval model can be chosen for this group of queries.

6.2 Background

This section introduces the background involved in this study. Some query features are introduced in section 6.2.1, and the correlation test methods applied in this chapter are described in section 6.2.2.

6.2.1 Statistical Query Features

In this section, we introduce some previously studied statistical query features.

- Query Length.

  Query length is the number of terms in a query. [He and Ounis, 2003, Zhai and Lafferty, 2004] show that query length has a strong effect on smoothing in

language modelling, and length normalization methods in probabilistic models. For short queries, the length impact is not significant.

- Distribution of Information Amount of Query Term.

  $idf(t)$ is usually viewed as a measure of the information carried by term t, and used to predict query performance. [He and Ounis, 2006] proposed variance $\sigma_{idf(t)}$ and $\frac{\max_{t\in q} idf(t)}{\min_{t\in q} idf(t)}$ to measure the distribution of information amount carried in a query, but their experiment shows that these two measures have a low correlation to retrieval performance.

- Query Scope.

  Query scope is the number of documents in the collection in which at least one query term occurs, and it measures the query specificity. Due to the sensitivity to collection size, the inverse algorithm $\log \frac{N_D(c)}{\max_{t\in q} n_D(t,c)}$ is applied. However, [He and Ounis, 2006] shows that there is no strong correlation between query scope and query performance.

- Query Clarity.

  Query clarity score is the measure of query ambiguity with respect to a document collection, initially proposed in [Cronen-Townsend et al., 2002]. It is the relative entropy between a query language model and a collection language model,which is expressed as follows $\sum_{t\in V} P(t|q)\cdot \log \frac{P(t|q)}{P(t|c)}$. [He and Ounis, 2006] simplify the query language model to the maximum likelihood estimation. Both of Cronen-Townsend and He find a strong correlation between query clarity score and MAP. However [He and Ounis, 2006]'s simplified clarity score does not outperform average inverse collection term frequency as query performance predictor, and the clarity score in [Cronen-Townsend et al., 2002] is expensive to compute.

- Average Inverse Document Frequency (AvIDF).

  Inverse document frequency is given by $\log \frac{N_D(c)}{n_D(t,c)}$, which is also an informativeness measurement for a query term. When used as query performance predictor, AvIDF shows some correlation to MAP, but not as strong as query clarity score [Cronen-Townsend et al., 2002].

- Average Inverse Collection Term Frequency (AvICTF).

Inverse collection term frequency, expressed as $\log \frac{N_L(c)}{n_L(t,c)}$, can be viewed as an alternative to IDF [Church and Gale, 1995]. It has strong correlation to average precision [He and Ounis, 2006]. [Wang and Roelleke, 2006] shows that inverse collection token frequency (ICTF) has similar retrieval performance to IDF.

So far, the correlation of AvgTF and performance, and its impact on model selection have not been studied. We will use AvgTF related features to predict which of the two models TF-IDF and LM works better on a query.

6.2.2 Ranking Correlation Test

In order to identify whether the retrieval result from two models are ranking correlated, we use some popular correlation methods in probability theory and statistics: Kendall, Pearson, and Spearman. These are used to test the strength and direction of the linear relationship between two random variables. The Pearson method tests the correlation of the value, The Kendall and Spearman test the correlation of the rank. However, the Spearman correlation is based on the rank assigned to the element of a list, whilst the Kendall coefficient is based on the rank order of the elements [Abdi, 2007].

The Kendall coefficient τ is used to study the ranking orders, and is calculated as follows:

$$\text{Kendall:} \quad \tau = \frac{\frac{1}{2}N(N-1) - D(P_1,P_2)}{\frac{1}{2}N(N-1)} \tag{6.1}$$

Here, N is the dimension of the variable, P is the set rank pairs, $D(P_1,P_2)$ is the number of discordant pairs.

The Pearson correlation is the product-moment correlation of the value of two random variables, which express the linear relation of two random variables.

$$\text{Pearson:} \quad r = \frac{N\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{N\sum x_i^2 - (\sum x_i)^2}\sqrt{N\sum y_i^2 - (\sum y_i)^2}} \tag{6.2}$$

Here x_i, y_i are two random variables.

Spearman correlation is the product-moment correlation coefficient for the rank:

$$\text{Spearman:} \quad \rho = 1 - \frac{6\Sigma d_i^2}{N(N^2-1)} \tag{6.3}$$

d_i is the rank difference between the two random variables x_i, y_i.

From the formula we can see that the denominator of τ and ρ are square or cubically increasing with regard to N. If N is very large then the coefficient will tend to be 1. This will be problematic when the correlation tests are based on high dimension data, or the data dimensions vary greatly, which is common in retrieval ranking lists. As for different queries, the sizes of retrieval results vary greatly, some queries have 100 retrieval result, whilst others have 100K retrieval result. The correlation coefficient may be very different. Even if the two retrieval models may behave very similarly on different queries, the correlation test result will be various due to the size of the retrieval result. Being aware of this shortcoming is important when interpreting the experiment results.

6.3 Average TF and TF Distribution

In this section, we will show the reason that we are particularly interested in AvgTF related query features as retrieval selectors. Previous research has successfully incorporated the AvgTF into the retrieval functions, whilst we believe not only AvgTF, but also the distribution of TF can impact the retrieval model performance. Because bursty terms tend to appear in a document with high term frequencies, while low term frequency for non-bursty terms. Indeed, when we looked into the detail of the TF distribution of query terms we found that the TF distributions are greatly different even if the query terms have the same AvgTFs. Here, we give the formal definition of AvgTF:

Definition 10 *Average term frequency (AvgTF) is defined as:*

$$avgtf(t,c) := \frac{n_L(t,c)}{n_D(t,c)} \tag{6.4}$$

In this section we use TREC-3 query terms as an example, showing AvgTF against DF in Figure 6.2. The horizontal and vertical lines in the figure are medians of AvgTF and DF, which respectively divide the query terms roughly into 4 groups. Figure 6.2 shows that some terms have high DF also high average TF, some have high DF and low average TF, some have low DF and high average TF, and the others have low DF and low average TF. Although the terms are grouped into four classes, it remains hard to show the TF distribution for each group. Figures 6.3, 6.4, 6.5, 6.6 show the TF distributions of some typical terms from each group, which can be very different even when the terms have similar AvgTFs and DFs.

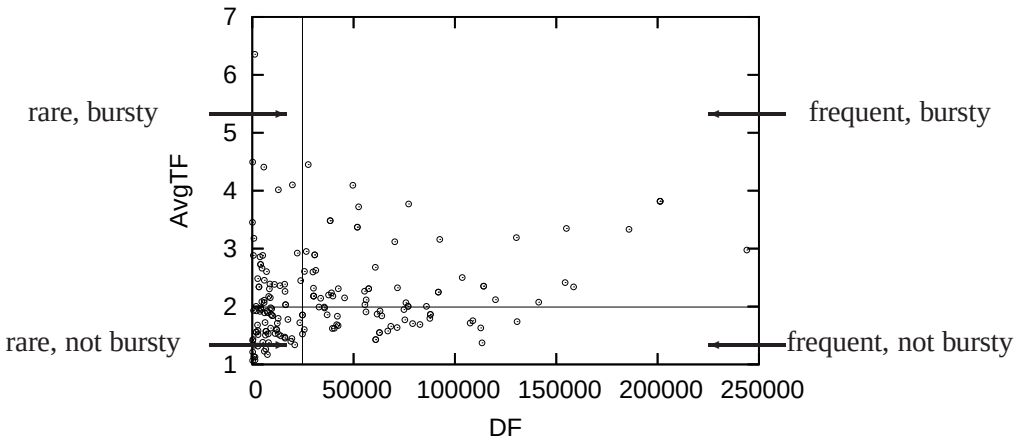
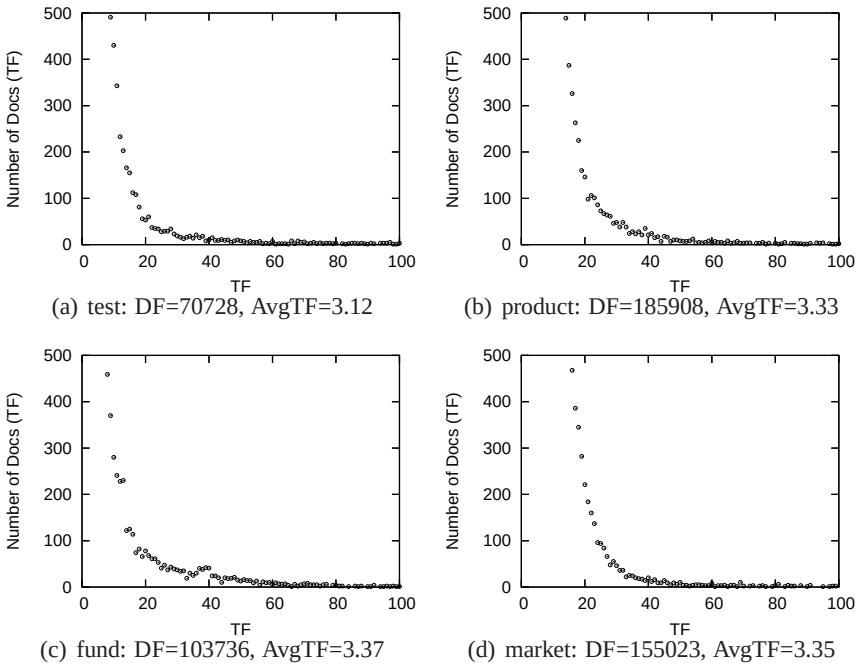


Figure 6.2: Average TF against DF: all 372 TREC-3 query terms



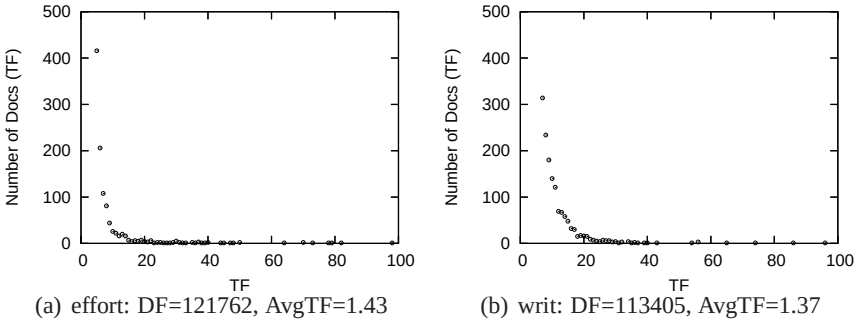
(a) test: DF=70728, AvgTF=3.12

(b) product: DF=185908, AvgTF=3.33

(c) fund: DF=103736, AvgTF=3.37

(d) market: DF=155023, AvgTF=3.35

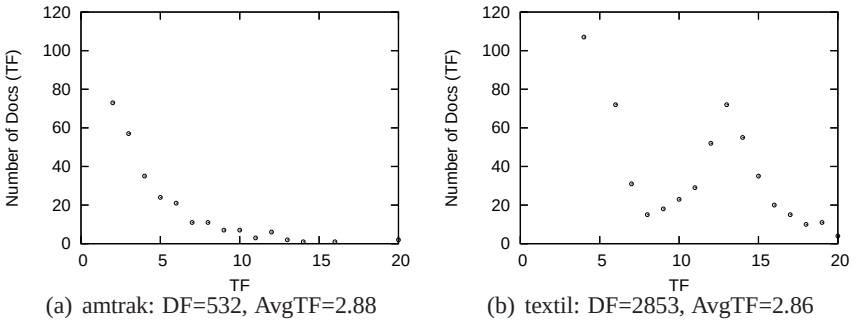
Figure 6.3: Number of Docs (TF): High DF and high average TF



(a) effort: DF=121762, AvgTF=1.43 (b) writ: DF=113405, AvgTF=1.37

(c) caus: DF=125480, AvgTF=1.55 (d) learn: DF=24782, AvgTF=1.52

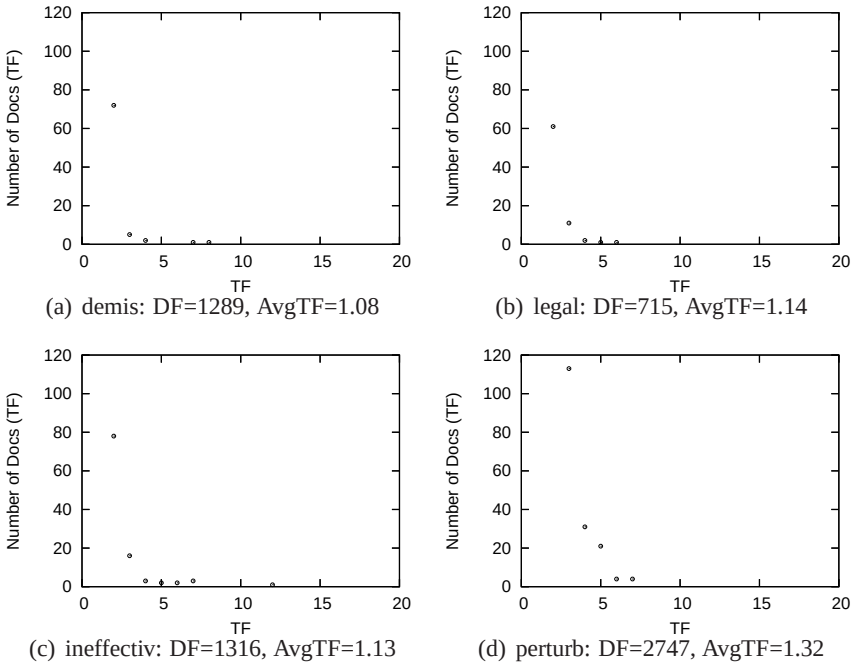
Figure 6.4: Number of Docs (TF): High DF and low average TF



(a) amtrak: DF=532, AvgTF=2.88 (b) textil: DF=2853, AvgTF=2.86

(c) asbesto: DF=1264, AvgTF=6.36 (d) nra: DF=209, AvgTF=4.49

Figure 6.5: Number of Docs (TF): Low DF and high average TF



(a) demis: DF=1289, AvgTF=1.08 (b) legal: DF=715, AvgTF=1.14

(c) ineffectiv: DF=1316, AvgTF=1.13 (d) perturb: DF=2747, AvgTF=1.32

Figure 6.6: Number of Docs (TF): Low DF and low average TF

Due to the big range of TF for some terms and the number of documents with a given TF, when we zoom the plot into a small area, some points will not be displayed. For example, high DF and AvgTF term "test", has more than 500 documents containing "test" only once ($|\{d|n_L(test,d)=1\}| >> 500$), the number of these documents will not be displayed as it is too large to fit in the figure. Therefore, the number of documents containing the term "test" more than 100 times ($|\{d|n_L(test,d)>100\}|$), and the high number for documents with very small TF ($|\{d|n_L(test,d)=1\}| >> 500$), will not be displayed in the figure either.

Intuitively we prefer the terms with high AvgTF and occurring in more documents with a TF greater than AvgTF, because this type of terms are the clinging terms. The number of documents that have a higher TF will more likely be high when the term's DF is high, therefore we think that the proportion of documents with a higher TF than AvgTF would be a more appropriate statistical query feature for model selection.

Next we will give the definition of the number and the percentage of documents with TF greater than AvgTF of a term t in definition 11 and 12.

Definition 11 *The number of documents with TF greater than AvgTF of a term t is defined as:*

$$\text{n-tf-gt-avgtf}(t,c) := |\{d|n_L(t,d) > avgtf(t,c)\}| \tag{6.5}$$

Definition 12 *Percentage of documents with TF greater than AvgTF of a term t:*

$$p\text{-}tf\text{-}gt\text{-}avgtf(t,c) := \frac{|\{d|n_L(t,d) > avgtf(t,c)\}|}{n_D(t,c)} \tag{6.6}$$

Here, $\{d|n_L(t,d) > avgtf(t,c)\}$ is a set of documents which contain term t with a frequency higher than the average term frequency $avgtf(t,c)$ in the whole collection. $n_D(t,c)$ is the number of documents that contain term t, and $n_L(t,d)$ is the times that term t occurs in the document d.

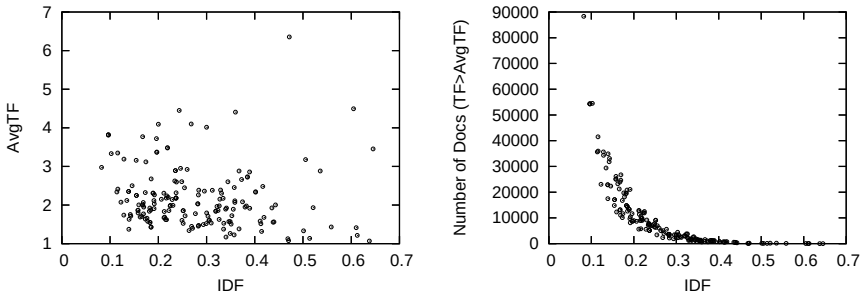
Figure 6.7 compares AvgTF, number and percentage of documents with TF above AvgTF to IDF. It shows that AvgTF, the number and percentage of documents with TF above AvgTF are all deviating from IDF. As IDF is not a very good query performance predictor (see [He and Ounis, 2006], [Cronen-Townsend et al., 2002]), so these AvgTF related statistical query features (SQF) may act as an alternative query performance predictor.

Even if some terms have the same AvgTF, the total number or percentage of documents containing a term with TF above AvgTF, the distribution of TF for these terms can be different. Let us look at figure 6.5, term "textil" has approximate 80 documents with TF=13, while other terms ("amtrak", "abesto" or "nra") have only few documents with TF=13. This distribution of query term may help to choose retrieval model for queries. [Church and Gale, 1995] observed that a good term is far from having a Poisson distribution: the number of documents containing term with high frequency is higher than the expected number of document based on Poisson distribution. Following this work, we expect that the query with more terms far from Poisson distribution will have better query performance. Therefore we also check the distance of the query term's observed TF distribution from Poisson distribution. The distance we use is defined in equation 6.7. It is also referred as relative entropy or Kullback-Leibler distance, and we will call it relative entropy in the rest of the chapter.

Definition 13 *Distance between observed TF distribution and Poisson TF distribution:*

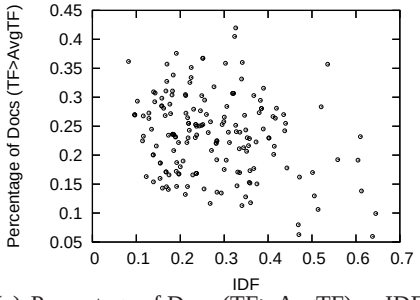
$$D(P_{observe}||P_{Poisson}) := \sum_{tf} P_{observe}(tf) \cdot \log \frac{P_{observe}(tf)}{P_{Poisson}(tf)} \tag{6.7}$$

In figure 6.8, we compare the relative entropy with IDF. It shows that a term with low IDF (high DF) tends to have high relative entropy in the document set containing this term, i.e. the relevant entropy is negatively correlated to IDF. The reason is that the size of document set containing each term is different. For the terms with the same or similar DF, their relative entropy



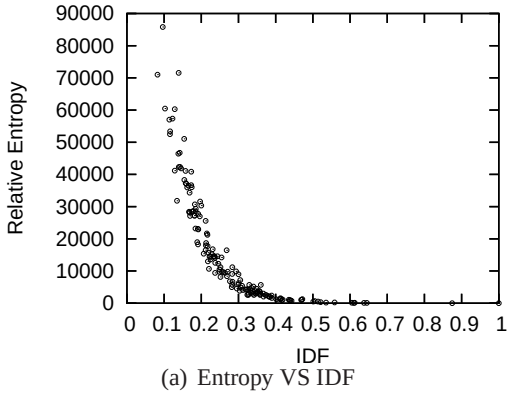
(a) AvgTF vs IDF

(b) Number of Docs (TF>AvgTF) vs IDF

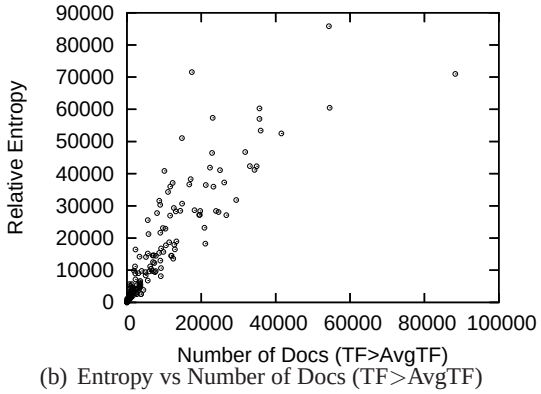


(c) Percentage of Docs (TF>AvgTF) vs IDF

Figure 6.7: AvgTF, number and percentage of documents (TF>AvgTF) vs normalized IDF



(a) Entropy VS IDF



(b) Entropy vs Number of Docs (TF>AvgTF)

Figure 6.8: Relative entropy vs IDF and number of documents (TF>AvgTF)

can be different. Because 6.8(a) is similar to figure 6.7(b), we plot the relative entropy against the number of documents with TF higher than AvgTF in figure 6.8(b). We can see that the number of documents that have TF above AvgTF is strongly correlated to the relative entropy ($coef_{Spearman} = 0.9450221, \quad p-value < 2.2e-16$). Therefore, the number of documents with TF above AvgTF can be used as an alternative to the relative entropy when considering the cost of computing.

In the later experiments, we will use the following AvgTF related query features to choose retrieval model for each query: AvgTF, the number and percentage of documents having query term with TF higher than AvgTF, and relevant entropy between the probability of observing a document containing query term t with TF n and probability of a document containing query term t with TF n expected based on Poisson distribution with an average TF m.

6.4 Mathematical Analysis of Ranking Functions

As there are many variations of ranking functions for LM and TF-IDF models, we give the definition of the ranking function for each retrieval model we use in this chapter in table 6.1.

Model	Retrieval Functions
TF-IDF	$RSV_{\text{TF-IDF}}(d,q,c) = \sum_{t \in d \cap q} tf_{\text{BM25}}(t,d) \cdot idf(t,c)$ (6.8)
	$tf_{\text{BM25}}(t,d) = \frac{n_L(t,d)}{n_L(t,d)+K} = P_L(t\|d), \quad K=1$ (6.9)
	$idf(t,c) = \log \frac{N_D(c)}{n_D(t,c)} = -\log P_D(t\|c)$ (6.10)
LM	$RSV_{\text{LM}}(d,q,c) = \sum_{t \in d \cap q} \log\left(1 + \frac{\lambda \cdot P_L(t\|d)}{(1-\lambda)\cdot P_L(t\|c)}\right)$ (6.11)
	$P_L(t\|d) = \frac{n_L(t,d)}{N_L(d)}, P_L(t\|c) = \frac{n_L(t,c)}{N_L(c)}, \lambda = 0.2$ (6.12)

Table 6.1: Two candidate retrieval models

Here, $n_L(t,x)$ is the number of locations (subscript L for location-based event space) at which term t occurs in x, where x can be a document, a query, the set of relevant or non-relevant documents, or the whole collection. Accordingly, $n_D(t,x)$ denotes the number of documents (subscript D for document-based event space). In TF-IDF model, K is set to 1, which leads to simplified BM25. And in LM, $\lambda = 0.2$ is applied, which is the general setting for the linear mixture model [Hiemstra, 2001].

Next, we will show how the term weights change with respect to TF and DF/LF. To make the formula shorter, we write the term weights as follows:

$$w_{\text{TF-IDF}}(tf_d, df) = \frac{tf_d}{tf_d+1} \cdot \log \frac{N_D}{df} \tag{6.13}$$

$$w_{\text{LM}}(tf_d, tf_c) = \log(1 + \frac{\lambda \cdot \frac{tf_d}{dl}}{(1-\lambda) \cdot \frac{tf_c}{cl}}) \tag{6.14}$$

Here $tf_d = n_L(t,d)$, $df = n_D(t,c)$, $tf_c = n_L(t,c)$, $dl = N_L(d)$, $cl = N_L(c)$, $N_D = N_D(c)$. Although we know that tf_d and tf_c/df are discrete variables in the retrieval system, the term weights all fall into the curve of the function we defined above. The partial derivative of the term function will show the changing rate of term with respect to DF and TF.

The partial derivative of the TF-IDF term weight is:

$$\frac{\partial w_{\text{TF-IDF}}}{\partial tf_d} = \frac{1}{(tf_d+1)^2} \cdot \log \frac{N_D}{df} \tag{6.15}$$

$$\frac{\partial w_{\text{TF-IDF}}}{\partial df} = -\frac{tf_d}{(tf_d+1) \cdot df} \tag{6.16}$$

Similarly the partial derivative of the LM term weight is:

$$\frac{\partial w_{\text{LM}}}{\partial tf_d} = \frac{1}{\frac{(1-\lambda) \cdot dl \cdot tf_c}{\lambda \cdot cl} + tf_d} \tag{6.17}$$

$$\frac{\partial w_{\text{LM}}}{\partial tf_c} = -\frac{\lambda \cdot cl \cdot tf_d}{(1-\lambda) \cdot dl \cdot tf_c^2 + \lambda \cdot tf_d \cdot cl \cdot tf_c} \tag{6.18}$$

From the partial derivatives, we can see that both of them are positive with respect to TF, they become smaller when TF increase (in other words, the term weight increasing rate will be less distinct with the increasing of TF). Partial derivatives with respect to DF/LF are both negative, and also the absolute values become smaller when DF/LF increase. In other words, the term weights become small with respect to the increasing of DF/LF, and the changing of term weights are less distinct when DF/LF are greater. To visualize the changing of term weights, we plot the term weights against TF and DF/LF in figure 6.9. To simplify the parameter in the figure, we use $P(t|d)$ and $P(t|c)$ instead of TF and DF/LF. The definitions of $P(t|d)$ and $P(t|c)$ used in the figure 6.9 are listed in table 6.2.

From the figure and partial derivative we can see that both TF-IDF and LM assign high weights to rare terms, especially TF-IDF will assign extremely high weights to rare terms.

TF-IDF	LM
$P_L(t\|d) = \frac{n_L(t,d)}{n_L(t,d)+1}$ $P_D(t\|c) = \frac{n_D(t,c)}{N_D(c)}$	$P_L(t\|d) = \frac{n_L(t,d)}{N_L(d)}$ $P_L(t\|c) = \frac{n_L(t,c)}{N_L(c)}$

Table 6.2: The definition of $P(t|d)$ and $P(t|c)$ for TF-IDF and LM

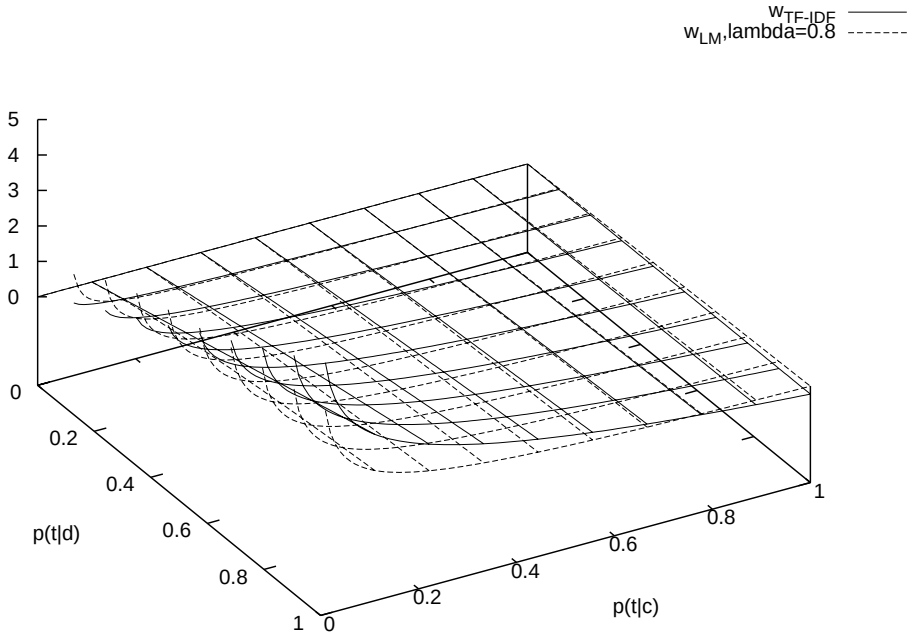


Figure 6.9: Term weights of TF-IDF and LM regarding $P_L(t|c)/P_D(t|c)$ and $P_L(t|d)$

Although it is possible to analyze the weight changing rate of a single term, it is not possible to predict multiple term weights as the term weight changing rate varies according to TF and DF, and the number of terms in a query is not fixed. Therefore we can not justify whether two models are equivalent or ranking equivalent mathematically, the proper way to test the ranking correlation of two models is to use statistical correlation test. Also there are no mathematical models between retrieval performance and SQFs, so again, we rely on statistical methods. In our experiments, we will use Spearman and Kendall ranking correlation to test the correlation between the ranking lists from TF-IDF and LM, and the correlation between SQF and the difference of APs from two models ($\delta_{AP} = AP_{\text{TF-IDF}} - AP_{\text{LM}}$).

6.5 Experiments and Results

The experiments are carried out on TREC-2, TREC-3 and TREC-8 collections with stemming and stop-word removal. TREC-2 and TREC-3 share the same document collection with 741,859 documents and 180,250,322 terms, but different queries; TREC-8 has 556,078 documents and 177,157,259 terms. TREC-3 is used to identify the query features that correlate to retrieval performance, and the thresholds that can divided the queries into two groups. Thereafter the criteria will be applied to other collections TREC-2 and TREC-8. The queries used in this chapter are title only, with an average query length 3.6 (max 6 and min 2) for TREC-3.

The experiment methodology is described in section 6.5.1. The results of the ranking correlation of the retrieval models and typical queries with high or low ranking correlations are presented in section 6.5.2. The correlations of SQFs and δ_{AP} are presented in section 6.5.3. In section 6.5.4, we use the criteria identified based on TREC-3 to choose retrieval model for the queries, also apply the criteria to TREC-2 and TREC-8.

6.5.1 Experimental Settings

The experiments in this study has three steps: ranking correlation of the retrieval models, the correlation of SQF and query performance, and model selection based on SQF.

- Ranking correlation of retrieval models.

  In order to test whether two retrieval models are ranking equivalent or highly correlated, we run each query with two retrieval models, then use statistical method to test the correlation of the ranking lists from the two models.

If the correlation coefficient is 1, then the two models are ranking equivalent. And the study should stop here, as the two models have the same retrieval performance. However this would not be the case. If there is no ranking equivalence exists, it is still interesting to find out whether some queries have high ranking correlation and what kind of features these query have.

Here, we compute the correlation coefficient based on the top 1000 retrieved documents. As stated previously, if we compute the correlation coefficient based on the whole ranking lists, the coefficient will be affected by the length of the ranking lists. For some queries with less discriminative terms, the ranking lists will be fairly long, whereas the queries with high discriminative terms retrieve relatively fewer documents. To better observe the ranking correlation of two retrieval models, we would also like to investigate how many overlaps between the ranking lists from two models.

Even if the two ranking lists are not highly correlated, they can have the same or very similar average precision. Therefore we also test the correlation of average precision for queries.

- Correlation of statistical query feature and retrieval performance.

  In this part of the experiment, we look into the correlation of a SQF and the difference of two models' AP ($\delta_{AP}, equals AP_{\text{LM}} - AP_{\text{TF-IDF}}$). The optimal case that we expect to observe is as what is shown in figure 6.1: 1) TF-IDF performs better with the queries having SQF greater than threshold v_1; 2) LM performs better with the queries having SQF lower than threshold v_2; 3) TF-IDF and LM have the same probability to performs better with queries having SQF between v_1 and v_2. As we apply TF-IDF to the queries in the group 3, therefore only the threshold v_1 needs to be found out.

  In order to find the threshold, we first plot down the statistical query features against δ_{AP}. If the result looks like the ideal example that we have given in figure 6.1, then it is easy to decide the threshold. Otherwise, the better method is to sum up the δ_{AP} from the query with smallest query feature, and find out the particular query feature value that make the $\sum \delta_{AP}$ to have the maximum value. This query feature value will be the threshold. The definition is as follows:

$$v = argmax(x) \sum_i (AP_{\text{LM}}(q_i) - AP_{\text{TF-IDF}}(q_i)), \quad sqf(q_i) < x \tag{6.19}$$

where i is query topic ID, $sqf(q_i)$ is the statistical query feature value for query q_i.

- Model selection based on SQF

  After identifying the threshold v_1 for each SQF, we can assign each query a better performing retrieval model according to the their SQF during the retrieval stage. If the query's SQF is less than v_1, then we apply LM to this query; otherwise we apply Tf-IDF to it.

  We firstly run the retrieval process with model selection on TREC-3, where we identified the threshold for the SQF, then we test this retrieval process on the other two test collection TREC-2 and TREC-8.

6.5.2 Ranking Correlation of TF-IDF and LM

In this section we show the ranking correlations of TF-IDF and LM for all 50 TREC-3 queries in table 6.3. All the correlation test are based on top 1000 retrieved documents. At the mean time, the overlaps of the retrieval results are shown in the table.

From table 6.3, we can see that the correlations of the ranking lists vary greatly from 0.856 to -0.104. And these ranking correlation coefficients have some correlation to the overlap of ranking lists ($coef_{Spearman} = 0.6434712, \quad p-value = 4.651e-07$). Generally if the query has high overlap from different models, then it tends to have high correlation between the ranking lists from the two models. Whereas some queries, like query 154, have high overlap but low ranking correlation. Most importantly, there is no high ranking correlation between TF-IDF and LM in general, which is necessary for model selection.

When we look into the ranking correlation for each query, we find that there are some queries that do have a very high or very low ranking correlation between the models. Thus we look into their statistical query features to see if these queries have the same SQF. We list in table 6.4 the top-5 and bottom-5 ranking correlated queries with DF and AvgTF of each query term. The queries that have highly correlated ranking lists usually have terms with small DFs, but no significant correlation to AvgTFs. However the correlation between $\sum_{t \in q} df(t)$ and the correlation coefficient for two ranking lists based on whole TREC-3 ad hoc topics, is not so strong ($coef_{Spearman} = -0.4441297, \quad p-value = 0.001371$).

QId	overlap @10	overlap @100	overlap @1000	Pearson	**Spearman**	Kendall
168	4	57	564	0.805	**0.856**	0.693
162	0	56	594	0.797	**0.828**	0.631
173	1	53	870	0.854	**0.823**	0.628
151	7	60	703	0.862	**0.819**	0.644
180	5	50	594	0.794	**0.815**	0.639
192	0	38	807	0.823	**0.783**	0.587
155	7	59	750	0.778	**0.766**	0.584
175	5	84	525	0.900	**0.751**	0.591
183	1	73	606	0.799	**0.747**	0.552
170	4	75	769	0.762	**0.743**	0.570
169	1	59	613	0.685	**0.727**	0.550
165	3	69	729	0.802	**0.725**	0.537
185	4	55	580	0.762	**0.699**	0.523
189	2	62	699	0.740	**0.689**	0.510
163	1	89	776	0.881	**0.684**	0.512
196	1	55	458	0.678	**0.616**	0.460
193	7	72	629	0.818	**0.611**	0.456
188	6	47	600	0.813	**0.588**	0.456
157	2	45	341	0.597	**0.587**	0.431
179	1	43	525	0.606	**0.570**	0.405
178	3	37	647	0.600	**0.541**	0.392
182	3	55	493	0.699	**0.538**	0.386
200	0	33	658	0.495	**0.535**	0.385
184	0	36	652	0.541	**0.521**	0.371
156	0	11	446	0.445	**0.521**	0.377
177	0	34	484	0.601	**0.501**	0.352
161	0	22	896	0.477	**0.461**	0.341
152	2	48	492	0.572	**0.459**	0.321
181	0	14	544	0.345	**0.436**	0.307
160	1	45	330	0.408	**0.436**	0.309
174	0	24	616	0.361	**0.424**	0.272
187	1	24	257	0.400	**0.421**	0.291
158	0	3	219	0.479	**0.416**	0.300
171	0	14	577	0.354	**0.383**	0.272
191	0	7	468	0.285	**0.381**	0.272
164	0	16	462	0.351	**0.373**	0.259
199	1	34	522	0.372	**0.364**	0.260
153	0	7	484	0.348	**0.359**	0.259
166	0	29	343	0.411	**0.354**	0.243
194	0	13	504	0.291	**0.306**	0.211
197	8	39	454	0.738	**0.298**	0.212
190	0	9	338	0.232	**0.297**	0.205
176	1	5	309	0.522	**0.275**	0.192
198	2	27	403	0.174	**0.263**	0.196
186	1	18	274	0.269	**0.223**	0.148
159	0	6	166	0.273	**0.177**	0.126
154	0	16	772	0.126	**0.167**	0.120
172	0	1	255	-0.012	**0.047**	0.033
195	1	6	492	0.142	**-0.039**	-0.014
167	0	5	294	-0.130	**-0.104**	-0.069

Table 6.3: Ranking correlation of TF-IDF and LM for each query: Sorted by Spearman ρ

QId	ql	overlap @1000	ρ	τ	DF	DF and AvgTF of query terms
168	2	564	0.856	0.693	42845	DF: amtrak=532, financ=42313 AvgTF: amtrak=2.8816, financ=1.6707
162	2	594	0.828	0.631	19268	DF: automobil=7985, recal=11283 AvgTF: automobil=1.3767, recal=1.5347
173	2	870	0.823	0.628	24649	DF: ban=17592, smok=705 AvgTF: ban=1.7768, smok=2.6031
151	3	703	0.819	0.644	62559	DF: cop=45557, overcrowd=847, prison=6155 AvgTF: cop=2.1499, overcrowd=1.3329, prison=2.2599
180	3	595	0.815	0.639	9074	DF: embargo=1942, ineffectiv=1316, sanct=5816 AvgTF: embargo=1.5633, ineffectiv=1.1261, sanct=2.0696
⋮						
159	3	166	0.177	0.126	287250	DF: car=91809, develop=158535, electr=36906 AvgTF: car=2.2493, develop=2.3371, electr=1.8560
154	2	772	0.167	0.120	42558	DF: oil=38477, spil=4081 AvgTF: oil=3.4824, spil=2.7263
172	3	255	0.047	0.033	271740	DF: effectiv=55594, med=30238, product=185908 AvgTF: effectiv=2.0313, med=2.1809, product=3.3322
195	5	492	-0.039	-0.014	250932	DF: attribut=16048, market=155023, perturb=2747, stock=77114 AvgTF: attribut=1.470, market=3.347, perturb=1.315, stock=3.770
167	4	294	-0.104	0.069	197863	DF: explicit=5175, regul=49668, show=107544, viol=35476 AvgTF: explicit=1.3813, regul=4.0920, show=1.7152, viol=1.9908

Table 6.4: TF-IDF and LM: Statistics for top-5 and bottom-5 correlated queries

Next, we list the queries with top and bottom retrieval performance from TF-IDF and LM separately in the tables 6.5 and 6.6. The two models share four common queries in top-5 performing queries, one in the bottom group. The average precisions in the bottom group are very low. Nevertheless, we can not draw a conclusion which type of queries perform well with respect to query length, IDF or AvgTF. Here, we test the correlation of average precision from two models. For the average precision, we care more about the value, therefore we use Pearson correlation to test the correlation of average precision for the two models, and we get $coef_{Pearson} = 0.8792603, \quad p-value < 2.2e-16$. This means for most queries, the two models behave the same, but for some queries they behave differently. If we are able to find out those queries with great δ_{AP} from two models based on statistical query feature, then we can improve the overall retrieval performance. In next section, we will show the correlation of SQF and the δ_{AP} from the two models.

QId	AP	P@10	ql	DF	DF and AvgTF of query terms
163	0.7321	0.9000	4	111027	DF: agen=92510, orang=4687, veteran=8665,vietnam=5165 AvgTF: agen=3.1569, orang=2.0841, veteran=2.3872,vietnam=2.8815
173	0.7214	0.7214	2	24649	DF: ban=17592, smok=7057 AvgTF: ban=1.7768 smok=2.6031
170	0.6651	0.8000	4	25927	DF: consequ=14053, gel=1865, implan=2656, silicon=7353 AvgTF: consequ=1.4928, gel=2.0032, implan=2.4819, silicon=1.9215
183	0.6538	0.9000	3	144244	DF: asbesto=1264, lawsuit=12326, rel=130654 AvgTF: asbesto=6.3552, lawsuit=1.7112, rel=1.7398
161	0.6070	0.6070	2	19803	DF: acid=10938, rain=8865 AvgTF: acid=2.3835, rain=2.1535
⋮					
186	0.0041	0.0000	4	200074	DF: differ=82754, inn=6624, learn=24782, level=85914 AvgTF: differ=1.6908, inn=1.5711, learn=1.5239, level=2.0018
194	0.0019	0.0000	3	189033	DF: earn=19542, money=56086, writ=113405 AvgTF: earn=1.4479, money=1.9054, writ=1.1.3736
187	0.0017	0.1000	4	196867	DF: demis=1289, independ=39540, publish=87649, sign=68389 AvgTF: demis=1.0784, independ=1.6213, publish=1.7944, sign=1.6581
167	0.0009	0.0000	4	197863	DF: explicit=5175, regul=49668, show=107544, viol=35476 AvgTF: explicit=1.3813, regul=4.0920, show=1.7152, viol=1.9908
172	0.0000	0.0000	3	271740	DF: effectiv=55594, med=30238, product=185908 AvgTF: effectiv=2.0313, med=2.1809, product=3.3322

Table 6.5: TF-IDF: Statistics for top-5 and bottom-5 performing query

QId	AP	P@10	ql	DF	DF and AvgTF of query terms
163	0.7935	0.9000	4	111027	DF: agen=92510, orang=4687, veteran=8665, vietnam=5165 AvgTF: agen=3.1569, orang=2.0841, veteran=2.3872,vietnam=2.8815
170	0.7380	0.9000	4	25927	DF: consequ=14053, gel=1865, implan=2656, silicon=7353 AvgTF: consequ=1.4928, gel=2.0032, implan=2.4819, silicon=1.9215
173	0.7342	0.7342	2	24649	DF: ban=17592, smok=705 AvgTF: ban=1.7768 smok=2.6031
183	0.6499	0.6499	3	144244	DF: asbesto=1264, lawsuit=12326, rel=130654 AvgTF: asbesto=6.3552, lawsuit=1.7112, rel=1.7398
151	0.5440	0.7000	3	62559	DF: cop=45557, overcrowd=847, prison=16155 AvgTF: cop=2.1499, overcrowd=1.3329, prison=2.2599
⋮					
195	0.0088	0.0000	4	250932	DF: attribut=16048, market=155023, perturb=2747, stock=77114 AvgTF: attribut=1.4707, market=3.3472, perturb=1.3151, stock=3.7703
186	0.0078	0.1000	4	200074	DF: differ=82754, inn=6624, learn=24782, level=85914 AvgTF: differ=1.6908, inn=1.5711, learn=1.5239, level=2.0018
194	0.0041	0.0000	3	189033	DF: earn=19542, money=56086, writ=113405 AvgTF: earn=1.4479, money=1.9054, writ=1.1.3736
155	0.0019	0.0000	4	90997	DF: christ=8075, fundamental=194, right=74808, wing=7920 AvgTF: christ=2.1736, fundamental=1.4124, right=1.9460, wing=1.5278
172	0.0000	0.0000	2	271740	DF: effectiv=55594, med=30238, product=185908 AvgTF: effectiv=2.0313, med=2.1809, product=3.3322

Table 6.6: LM: Statistics for top-5 and bottom-5 performing query

6.5.3 Correlations of SQFs and δ_{AP} of TF-IDF and LM

In table 6.7, we list the average precisions from two models, δ_{AP} ($AP_{TF-IDF} - AP_{LM}$) and the statistical query features (DF, AvgTF and Entropy) for each TREC-3 query, also the correlations of SQFS and δ_{AP}. Out of 50 query topics, eight queries have better results from TF-IDF model, and LM generally has better retrieval performance. Details of these 8 queries are listed in table 6.8. The correlation of δ_{AP} with some statistical query features are also listed the result on the last two rows of table 6.7. The correlation between relative entropy and δ_{AP} is the strongest one among all the query features we tested.

Table 6.8 shows that the queries having better TF-IDF performance usually have small DFs. However, there is no conclusion for AvgTF, and these AvgTFs of query terms range from 1.1716 to 3.4824 compared to average AvgTF 2.1422.

We obtain the threshold by plotting the SQFs against the δ_{AP} ($AP_{\text{LM}} - AP_{\text{TF-IDF}}$) in figure 6.10, and observing the threshold which can divide the queries into two groups. In one group, most of the queries have better performance with LM, in the other group, most have better performance with TF-IDF. The observed thresholds are 6 for AvgTF, 20,000 for relative entropy, 20,000 for number of documents with TF higher than AvgTF and 0.6 for percentage of docs with TF above AvgTF. The threshold can also be automatically calculated according to the formula 6.19.

Next we will run model selection experiment based on the threshold identified in this section.

6.5.4 Experiment Results of Model Selection

With the thresholds identified in the previous section, we choose a retrieval model for each query. If the SQF of a query is less than the threshold, then LM is applied, otherwise, TF-IDF is applied. Firstly we run the model selection retrieval on TREC-3, and evaluate the result in table 6.9. The best run is **MAP=0.2252 and P@10=0.4460**, by using relative entropy.

To test whether the SQFs and their threshold identified in one collection will work on the other collections, we run the experiments in the same way, but on different test collection: TREC-2 (same document collection as TREC-3) and TREC-8 (different document collection). We obtain the results in table 6.10. Both of them have a minor improvement with model selection.

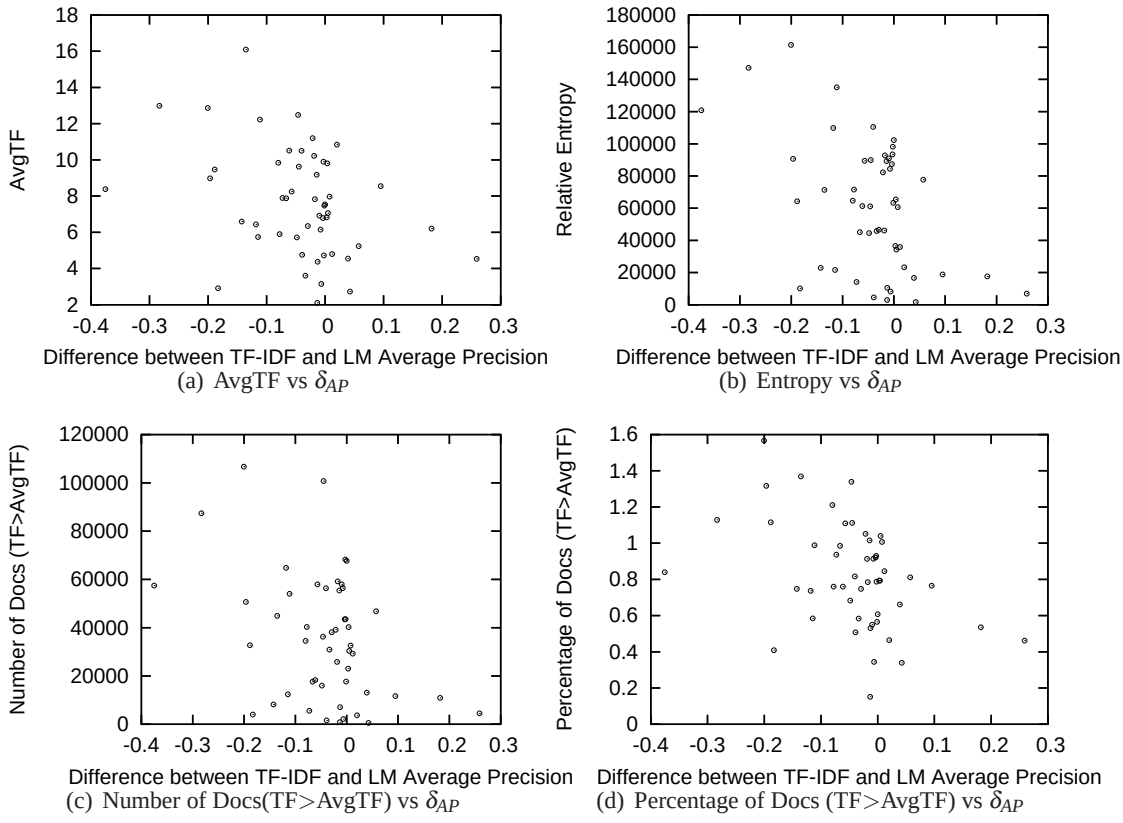
To better understand why the criteria found in TREC-3 failed to work in the other two collections, we looked into the correlation of SQF and δ_{AP} on TREC-2,3,8 in table 6.11. It shows

QId	TF-IDF		LM		δAP	δP10	DF	AvgTF	Entropy
	AP	P@10	AP	P@10					
161	0.607	0.607	0.348	0.500	**0.259**	0.107	19803	4.54	6918.0
154	0.538	0.100	0.357	0.100	**0.182**	0.000	42558	6.21	17648.4
192	0.408	0.600	0.313	0.100	**0.095**	0.500	45806	8.55	18841.9
160	0.242	0.300	0.185	0.400	**0.057**	-0.100	172266	5.24	77741.3
188	0.194	0.900	0.151	0.900	**0.043**	0.000	2990	2.73	1671.5
168	0.281	0.600	0.241	0.200	**0.039**	0.400	42845	4.55	16665.0
174	0.247	0.200	0.226	0.100	**0.020**	0.100	28753	10.85	23324.0
189	0.145	0.600	0.133	0.300	**0.012**	0.300	92056	4.80	35906.0
169	0.141	0.200	0.133	0.100	**0.008**	0.100	140547	7.97	60652.2
155	0.007	0.000	0.002	0.000	**0.005**	0.000	90997	7.06	34273.6
183	0.654	0.900	0.650	0.650	**0.004**	0.250	144244	9.81	65417.0
179	0.022	0.000	0.019	0.100	**0.003**	-0.100	96334	6.83	36541.0
172	0.000	0.000	0.000	0.000	**0.000**	0.000	271740	7.54	102335.6
171	0.018	0.000	0.019	0.000	**-0.001**	0.000	107152	7.48	63398.2
194	0.002	0.000	0.004	0.000	**-0.002**	0.000	189033	4.73	98213.9
195	0.006	0.000	0.009	0.000	**-0.003**	0.000	250932	9.90	93422.0
186	0.004	0.000	0.008	0.100	**-0.004**	-0.100	200074	6.79	87433.9
193	0.215	0.215	0.222	0.222	**-0.006**	-0.006	12708	3.16	8220.8
187	0.002	0.100	0.009	0.100	**-0.008**	0.000	196867	6.15	84447.1
197	0.011	0.400	0.020	0.400	**-0.010**	0.000	217892	6.92	90939.6
178	0.063	0.000	0.076	0.000	**-0.013**	0.000	5816	2.11	2973.5
173	0.721	0.721	0.734	0.734	**-0.013**	-0.013	24649	4.38	10531.3
167	0.001	0.000	0.015	0.000	**-0.014**	0.000	197863	9.18	89289.5
185	0.258	0.400	0.275	0.800	**-0.018**	-0.400	221971	7.83	92831.5
200	0.133	0.100	0.152	0.400	**-0.019**	-0.300	117249	10.23	46147.5
152	0.057	0.300	0.079	0.500	**-0.021**	-0.200	179105	11.20	82284.9
177	0.153	0.100	0.182	0.500	**-0.029**	-0.400	154847	6.35	46552.6
176	0.014	0.000	0.047	0.300	**-0.034**	-0.300	104651	3.61	45884.2
180	0.180	0.400	0.220	0.300	**-0.039**	0.100	9074	4.76	4540.8
184	0.034	0.000	0.074	0.300	**-0.040**	-0.300	267530	10.50	110483.3
165	0.289	0.700	0.334	0.700	**-0.045**	0.000	290485	9.63	89876.5
175	0.174	0.300	0.220	0.300	**-0.046**	0.000	153400	12.48	61108.9
198	0.032	0.000	0.080	0.200	**-0.048**	-0.200	84104	5.72	44544.2
190	0.005	0.000	0.062	0.300	**-0.057**	-0.300	226996	8.26	89435.1
163	0.732	0.900	0.793	0.900	**-0.061**	0.000	111027	10.51	61341.6
166	0.034	0.000	0.100	0.300	**-0.067**	-0.300	91884	7.88	45118.3
170	0.665	0.800	0.738	0.900	**-0.073**	-0.100	25927	7.90	14174.8
191	0.059	0.200	0.137	0.400	**-0.078**	-0.200	158543	5.90	71596.5
181	0.015	0.000	0.095	0.100	**-0.080**	-0.100	158653	9.85	64626.1
164	0.009	0.000	0.121	0.500	**-0.111**	-0.500	314007	12.23	135117.7
151	0.429	0.600	0.544	0.700	**-0.115**	-0.100	62559	5.74	21630.4
159	0.013	0.000	0.132	0.500	**-0.118**	-0.500	287250	6.44	109822.6
182	0.171	0.600	0.306	0.800	**-0.135**	-0.200	162512	16.10	71365.8
199	0.034	0.000	0.176	0.500	**-0.143**	-0.500	45583	6.60	22886.4
162	0.226	0.400	0.409	0.409	**-0.183**	-0.009	19268	2.91	10138.3
157	0.215	0.400	0.404	0.700	**-0.189**	-0.300	140620	9.47	64275.5
156	0.053	0.200	0.249	0.900	**-0.197**	-0.700	216153	8.98	90677.9
153	0.023	0.000	0.223	0.600	**-0.200**	-0.600	389192	12.87	161481.1
196	0.247	0.100	0.530	0.900	**-0.283**	-0.800	331008	12.99	147246.3
158	0.034	0.000	0.409	0.800	**-0.375**	-0.800	274018	8.39	120854.4
cor ρ							-0.3656	-0.3550	-0.4842
p-value							0.0094	0.0118	0.0004

Table 6.7: TF-IDF and LM: AP and P@10 for each query, correlation of SQFs and δ_{AP}

QId	AP_{TF-IDF}	AP_{LM}	DF	DF and AvgTF of query terms
161	0.6070	0.3481	19803	DF: acid=10938, rain=8865 AvgTF: acid=2.3835, rain=2.1535
154	0.5385	0.3566	42558	DF:oil=38477, spil=4081 AvgTF: oil=3.4824, spil=2.7263
192	0.4080	0.3130	45806	DF: cleanup=3248, oil=38477, spil=4081 AvgTF: oil=3.4824, spil=2.7263, cleanup=2.3396
160	0.2421	0.1848	172266	DF: caus=62740, cur=108877, vitamin=649 AvgTF: vitamin=1.9322, cur=1.7542, caus=1.5506
188	0.1941	0.1515	2990	DF: beachfront=188, eros=2802 AvgTF: beachfront=1.2128, eros=1.5157
168	0.2807	0.2415	42845	DF:amtrak=532, financ=42313 AvgTF: financ=1.6706, amtrak=2.8816
189	0.1448	0.1329	92056	DF: motiv=7446, murd=9318, real=75292 AvgTF: real=1.7702, motiv=1.1716, murd=1.8592
169	0.1410	0.1334	140547	DF: cost=103601, garbag=2059, remov=32893, trash=1994 AvgTF: cost=2.5, remov=1.9913, garbag=1.9262, trash=1.5572

Table 6.8: Statistics for top queries with better TF-IDF performance than LM



(a) AvgTF vs δ_{AP} (b) Entropy vs δ_{AP}

(c) Number of Docs(TF>AvgTF) vs δ_{AP} (d) Percentage of Docs (TF>AvgTF) vs δ_{AP}

Figure 6.10: Statistical query features vs δ_{AP} ($AP_{\text{TF-IDF}} - AP_{\text{LM}}$)

that the correlations between SQF and δ_{AP} on TREC-2 and TREC-8 are much lower than TREC-3, although the correlation in TREC-3 is already relative low. This explains why there is no distinguishable improvement on retrieval performance in the other two collections. This also underlines that the SQF and δ_{AP} need to be correlated in order to use a SQF to choose a model which is likely to perform best.

	TREC-3					
	MAP	Improvement%		P@10	Improvement%	
		TF-IDF	LM		TF-IDF	LM
TF-IDF	0.1763			0.2880		
LM	0.2194			0.4300		
AvgTF=6	0.2169	23.0	-1.1	0.4260	47.9	-0.9
Entropy=20000	**0.2252**	27.7	2.6	**0.4460**	54.9	3.7
Number of Docs=20000	0.2169	23.0	-1.1	0.4260	47.9	-0.9
Percentage of Docs=0.6	0.2212	25.5	0.8	0.4240	47.2	-1.4

Table 6.9: Using AvgTF related SQF to choose different models in TREC-3

	TREC-2					
	MAP	Improvement%		P@10	Improvement%	
		TF-IDF	LM		TF-IDF	LM
TF-IDF	0.1641			0.3260		
LM	0.1541			0.3560		
AvgTF=6	0.1621	-1.2	5.2	0.3500	7.4	-0.17
Entropy=20000	0.1652	0.8	7.2	0.3620	11.0	1.7
Number of Docs=20000	0.1263	23.0	-18.0	0.3294	1.0	-7.5
percentage of Docs=0.6	**0.1837**	11.9	19.2	**0.4370**	34.0	22.8
	TREC-8					
	MAP	Improvement%		P@10	Improvement%	
		TF-IDF	LM		TF-IDF	LM
TF-IDF	0.1955			0.3260		
LM	0.2323			0.4220		
AvgTF=6	0.2343	19.8	0.9	0.4320	32.5	2.4
Entropy=20000	0.2240	14.6	-3.6	0.4080	25.2	-3.3
Number of Docs=20000	0.2126	8.7	-8.5	0.3740	14.7	-17.8
Percentage of Docs=0.6	**0.2348**	20.1	1.1	**0.4360**	33.7	3.3

Table 6.10: Using AvgTF related SQF to choose different models in TREC-2 and TREC-8

6.6 Summary

This chapter studied the ranking correlation of retrieval models and how to select an appropriate model for a query according to its statistical query feature (SQF).

	AvgTF		Entropy		Number of Docs (TF>AvgTF)		Percentage of Docs (TF>AvgTF)	
	ρ	p-value	ρ	p-value	ρ	p-value	ρ	p-value
TREC2	-0.298	0.035	-0.240	0.091	-0.251	0.077	-0.102	0.478
TREC3	-0.359	0.010	-0.366	0.009	-0.406	0.003	-0.2981	0.035
TREC8	-0.096	0.502	-0.079	0.583	-0.156	0.278	-0.0095	0.947

Table 6.11: Correlations of δ_{AP} and SQF in TREC-2,3,8

Regarding SQFs, this study focused on AvgTF, TF distribution, DF, and relative entropy as criteria for model selection, as it is known that including AvgTF into a retrieval model can improve retrieval performance and combining AvgTF and IDF can predict query performance to a certain degree. The main findings regarding these particular SQFs are: (1) Only when there exists correlation between a SQF and δ_{AP}, it will be possible to improve the retrieval performance by SQF-based model selection; (2) the relative entropy of observed TF and random (Poisson-expected) TF are more correlated to δ_{AP} than AvgTF and DF are.

The main contribution of the study with SQF and ranking correlation is the methodology for model selection based on statistical query features: investigate the correlation of models and the correlation of SQF and performance difference in order to decide whether and how to select a model. The work focuses on TF-IDF and LM, and a few SQFs. Currently the model selection is based on one single query feature, and this could be potentially extended into a combined feature space. This study confirms that it is difficult to identify a sharp threshold for a single SQF. The correlations between SQFs and δ_{AP} differ in different collections; therefore, future work is to incorporate the correlation coefficient into the performance prediction (model selection). Also, the model selection should have access to possible a large amount of models, i.e. the space of models contains many synthetic variants of TF-IDF, LM and other models where parameter learning aims at both, firstly at the optimization of the retrieval quality, and secondly at achieving a strong correlation between a SQF and performance difference (δ_{AP}) to enable model selection.

Chapter 7

Conclusions

This chapter summarises the work that has been carried out in this thesis. We also outline our contributions to IR and make suggestions for future research.

7.1 Summary

Firstly, we investigated the theoretical interpretation of TF-IDF from various aspects based on binary independence retrieval (BIR) model, Poisson model, language modelling (LM), and information theory. With the "average" characteristic of the Poisson model, we built a bridge between BIR model and LM, and showed that the components of TF-IDF are an inherent part of this equation. Based on the term independence assumption, we showed that the decomposition of relevance probabilities or odds, which are adopted in the classical probabilistic model BIR model and language modelling, will lead to TF-IDF formulation under certain assumptions. While the decomposition based on the term disjointness assumption leads to the DQI model, which measures the relevance of a document to a query according the dependency between the document and the query. Once more the DQI model theoretically justifies TF-IDF: TF-IDF is the integral over the DQI model.

Secondly, we introduced an important aspect of this thesis, "probabilistic logical modelling". We illustrated how to build retrieval models with the probabilistic standard query language and relational algebra, namely PSQL and PRA. We demonstrated the different ways of probability aggregation with the TF-IDF model, we also demonstrated that different probabilistic assumptions in the modelling can result in the same model having different results. For non-probabilistic

operations, we presented alternative ways, some of which improved retrieval performance. In this thesis, we implemented a classical probabilistic model BIR model, language modelling, a non-probabilistic model TF-IDF, and a precision- and recall-based evaluation.

Thirdly, we proposed context-specific frequencies for probability estimation and term weighting in structured document retrieval. These frequencies were based on the ranking requirement that the parent element should be ranked higher if most of its children contain the query term, or ranked lower if very few of its children contain the query term. The context-specific IDF helps to meet such a requirement when it is defined as a generalised inverse element frequency. The element retrieval with context-specific frequencies did not meet our original expectations. However, context-specific frequencies showed promising results in our mock distributed retrieval in the hierarchically structured collections, where each sub-collection did not have a common subject. In such an environment, document retrieval with context-specific frequencies showed similar performance to the retrieval using global information. This discovery has a practical implication to distributed retrieval.

Finally, we presented a method how to choose a query a retrieval model that is likely to perform well, based on the correlation of statistical query features and model performance. This study was motivated by the observation that there is not a single retrieval model which can outperform all other models on all queries. For some queries, one model performs better and for another set of queries, a different model is superior. This observation suggested that retrieval performance could be improved by choosing an appropriate retrieval model for each query rather than elaborating a single model for all of the queries. Based on this postulation, we chose TF-IDF and LM as our candidate retrieval models. Then, we examined the ranking correlation of the two models and the correlation between the retrieval performance and TF related statistical query features. Unfortunately, it remained difficult to find a sharp threshold with single feature which would group the queries for a specific retrieval model. Nevertheless, the retrieval result from the model selection framework can behave differently when the feature space and candidate models are extended to a larger scale.

7.2 Contribution and Future Work

In this section, we outline the contribution of and future work for each aspect of this thesis:

- The strictly parallel investigation of BIR model, Poisson model and LM clarifies the event

spaces, the ranking rationales and the relevance assumptions for the aforementioned models. Moreover, the derivation demonstrates that the Poisson model can be viewed as a bridge (Poisson bridge) connecting BIR and LM. Therefore, BIR and LM can be used in a dual way to represent TF-IDF. Further study of the LM-like decomposition of $P(d|q)$ for independent terms yields a TF-IDF interpretation which is related to the probabilistic odds $O(r|d,q)$. On the other hand, the decomposition of $P(q|d)$ based on disjoint terms leads to a document and query independence(DQI) model which can be related to TF-IDF by using the integral.

In the future, the retrieval quality of the DQI model can be investigated. As this model shows similar formulation to mutual information, we are tempted to explain the DQI model as the mutual information between a query and a document. Thus, further investigation on the DQI model with mutual information theory is worthwhile.

- Probabilistic relational modelling enables both probability estimation and quality measurement to be integrated into the retrieval model. Such a mechanism gives the modelling greater flexibility, allowing probability estimation to be conducted during retrieval time without instantiated probability representation being required. Moreover, probabilistic relational modelling facilitates implementing retrieval models in a compact and flexible way in order to meet customised user information needs. Such characteristics are useful and desirable in order to support the development of ranking strategies beyond the classical document retrieval.

  In the future, how to incorporate complex probability estimation needs to be investigated in order to be able to implement more ranking models. Furthermore, the probabilistic relational modelling can be applied to other searching corpora rather than just document-based corpora, e.g. web-mining or product recommendation.

- The context-specific frequencies can bring document and element retrieval into a generalised retrieval model, even the collection selection. Context-specific discriminativeness can be understood as choosing the right discriminativeness space for different retrieval objects. Our experiments show that context-specific frequencies can help to effectively rank documents in a structured document collection, despite the fact that they do not perform well with element ranking. The results indicate that context-specific frequencies can bring

scalability and flexibility to retrieval management. Applying the context-specific frequencies to the distributed retrieval environment avoids both maintaining centralised statistical information and using global information to rank documents.

Additional tests regarding context-specific frequencies on collections without a general topic would be helpful to further validate the effectiveness of context-specific frequencies. Since the context-specific discriminativeness from a single context does not achieve good retrieval performance in element retrieval, combining term frequencies in a certain context with frequencies in the parental context for term weighting should be considered for element retrieval in the future.

- The study of model selection contributes a methodology to select the appropriate model based on both the ranking correlation of retrieval models, and the correlation of statistical query features with model performance.

  In future model selection work, the correlation coefficient can be incorporated into a performance predicting function to deal with the fact that the correlations of statistical query features and query performance vary according the collection. Machine learning techniques such as maximum entropy or Bayesian neural networks can be applied for query classification. When a single feature is insufficient for model selection, multiple features can be applied. Therefore, more query features are needed to be identified and the model selection should have access to a large number of models.

Bibliography

[Abdi, 2007] Abdi, H. (2007). Kendall rank correlation. *Encyclopedia of Measurement and Statistics*.

[Agrawal et al., 2003] Agrawal, S., Chaudhuri, S., Das, G., and Gionis, A. (2003). Automated ranking of database query results. In *CIDR*.

[Aizawa, 2003] Aizawa, A. (2003). An information-theoretic perspective of tf-idf measures. *Information Processing and Management*, 39:45–65.

[Amati et al., 2004] Amati, G., Carpineto, C., and Romano, G. (2004). Query difficulty, robustness, and selective application of query expansion. In *ECIR*, pages 127–137.

[Amati and van Rijsbergen, 2002] Amati, G. and van Rijsbergen, C. J. (2002). Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transaction on Information Systems (TOIS)*, 20(4):357–389.

[Anh and Moffat, 2002] Anh, V. N. and Moffat, A. (2002). Vector space ranking: Can we keep it simple? In *ADCS*.

[Azzopardi, 2005] Azzopardi, L. (2005). *Incorporating Context within the Language Modeling Approach for ad hoc Information Retrieval*. PhD thesis, University of Paisley.

[Azzopardi and Roelleke, 2007] Azzopardi, L. and Roelleke, T. (2007). Explicitly considering relevance within the language modelling framework. In *Proceedings of the 1st International Conference on Theory of Information Retrieval (ICTIR 07) - Studies in Theory of Information Retrieval*.

[Baeza-Yates and Ribeiro-Neto, 1999] Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison Wesley.

[Betsi et al., 2006] Betsi, S., Lalmas, M., Tombros, A., and Tsikrika, T. (2006). User expectations from xml element retrieval. In *Proceedings of the 29th annual international ACM SIGIR*

conference on Research and development in information retrieval, SIGIR '06, pages 611–612, New York, NY, USA. ACM.

[Blanco and Barreiro, 2008] Blanco, R. and Barreiro, A. (2008). Probabilistic document length priors for language models. In *ECIR*, pages 394–405.

[Bookstein, 1980] Bookstein, A. (1980). Fuzzy requests: An approach to weighted Boolean searches. *Journal of the American Society for Information Science*, 31:240–247.

[Brin and Page, 1998] Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117.

[Callan et al., 2001] Callan, J., Connell, M., and Connell, N. M. (2001). Query-based sampling of text databases. *ACM Transactions on Information Systems*, 19:2001.

[Callan et al., 1995a] Callan, J., Lu, Z., and Croft, W. (1995a). Searching distributed collections with inference networks. In Fox, E., Ingwersen, P., and Fidel, R., editors, *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–29, New York. ACM.

[Callan, 1994] Callan, J. P. (1994). Passage-level evidence in document retrieval. In [Croft and van Rijsbergen, 1994], pages 302–310.

[Callan et al., 1995b] Callan, J. P., Lu, Z., and Croft, W. B. (1995b). Searching distributed collections with inference networks. In *SIGIR '95: Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 21–28, New York, NY, USA. ACM.

[Chandel et al., 2007] Chandel, A., Hassanzadeh, O., Koudas, N., Sadoghi, M., and Srivastava, D. (2007). Benchmarking declarative approximate selection predicates. In *SIGMOD Conference*, pages 353–364.

[Chaudhuri et al., 2006] Chaudhuri, S., Das, G., Hristidis, V., and Weikum, G. (2006). Probabilistic information retrieval approach for ranking of database query results. *ACM Trans. Database Syst.*, 31(3):1134–1168.

[Chen and Goodman, 1996] Chen, S. F. and Goodman, J. (1996). An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Associ-*

ation for Computational Linguistics, pages 310–318, Morristown, NJ, USA. Association for Computational Linguistics.

[Chen et al., 2000] Chen, S. F., Rosenfeld, R., and Member, A. (2000). A survey of smoothing techniques for me models. *IEEE Transactions on Speech and Audio Processing*, 8:37–50.

[Church and Gale, 1995] Church, K. and Gale, W. (1995). Inverse document frequency (idf): A measure of deviation from poisson. In *Proceedings of the Third Workshop on Very Large Corpora*, pages 121–130.

[Church and Hanks, 1990] Church, K. W. and Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

[Cohen, 1998] Cohen, W. W. (1998). Integration of heterogeneous databases without common domains using queries based on textual similarity. In Haas, L. M. and Tiwary, A., editors, *SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA*, pages 201–212. ACM Press.

[Crestani et al., 1998] Crestani, F., Lalmas, M., van Rijsbergen, C., and Campbell, I. (1998). "is this document relevant?...probably". a survey of probabilistic models in information retrieval. *Computing Surveys*, 30.

[Croft and Harper, 1979] Croft, W. and Harper, D. (1979). Using probabilistic models of document retrieval without relevance information. *Journal of Documentation*, 35:285–295.

[Croft et al., 1998] Croft, W. B., Moffat, A., van Rijsbergen, C. J., Wilkinson, R., and Zobel, J., editors (1998). *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York. ACM.

[Croft and van Rijsbergen, 1994] Croft, W. B. and van Rijsbergen, C. J., editors (1994). *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, London, et al. Springer-Verlag.

[Cronen-Townsend et al., 2002] Cronen-Townsend, S., Zhou, Y., and Croft, W. B. (2002). Predicting query performance. In *Proceedings of the 25th Annual International ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR 2002)*, pages 299–306.

[de Vries and Roelleke, 2005] de Vries, A. and Roelleke, T. (2005). Relevance information: A loss of entropy but a gain for IDF? In *ACM SIGIR*, pages 282–289, Salvador, Brazil.

[Deerwester et al., 1990] Deerwester, S., Dumais, S., Furnas, G., Landauer, T., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

[Dumais et al., 1988] Dumais, S. T., Furnas, G. W.and Landauer, T. K., and Deerwester, S. (1988). Using latent semantic analysis to improve information retrieval. pages 281–285.

[Fuhr, 1992] Fuhr, N. (1992). Probabilistic models in information retrieval. *Comput. J.*, 35(3):243–255.

[Fuhr and Buckley, 1991] Fuhr, N. and Buckley, C. (1991). A probabilistic learning approach for document indexing. *ACM Transactions on Information Systems*, 9(3):223–248.

[Fuhr et al., 2003a] Fuhr, N., Goevert, N., Kazai, G., and Lalmas, M., editors (2003a). *INEX: Evaluation Initiative for XML retrieval - INEX 2002 Workshop Proceedings*, DELOS Workshop.

[Fuhr et al., 2003b] Fuhr, N., Goevert, N., Kazai, G., and Lalmas, M., editors (2003b). *INEX: Evaluation Initiative for XML retrieval - INEX 2002 Workshop Proceedings*.

[Fuhr and Großjohann, 2001] Fuhr, N. and Großjohann, K. (2001). XIRQL: A query language for information retrieval in XML documents. pages 172–180.

[Fuhr and Roelleke, 1997] Fuhr, N. and Roelleke, T. (1997). A probabilistic relational algebra for the integration of information retrieval and database systems. *ACM Transactions on Information Systems*, 14(1):32–66.

[Gale and Church, 1991] Gale, W. A. and Church, K. W. (1991). Identifying word correspondences in parallel texts. In *HLT*.

[Generet et al., 1995] Generet, M., Ney, H., and Wessel, F. (1995). Extensions of absolute discounting for language modeling. In *In EUROSPEECH-1995*.

[Grabs and Schek, 2002] Grabs, T. and Schek, H.-J. (2002). Generating vector spaces on-the-fly for flexible xml retrieval. In *Proceedings of the ACM SIGIR Workshop on XML and Information Retrieval*, pages 4–13, Tampere, Finland. ACM.

[Gravano and Garcia-Molina, 1995] Gravano, L. and Garcia-Molina, H. (1995). Generalizing gloss to vector-space databases and broker hierarchies. Technical report, Stanford, CA, USA.

[Greiff, 1998] Greiff, W. (1998). A theory of term weighting based on exploratory data analysis. In [Croft et al., 1998], pages 11–19.

[Harter, 1975a] Harter, S. (1975a). A probabilistic approach to automatic keyword indexing. part I: On the distribution of speciality words in a technical literature. *Journal of the American Society for Information Science*, 26:197–206.

[Harter, 1975b] Harter, S. (1975b). A probabilistic approach to automatic keyword indexing. part II: An algorithm for probabilistic indexing. *Journal of the American Society for Information Science*, 26:280–289.

[Hawking and Thistlewaite, 1999] Hawking, D. and Thistlewaite, P. (1999). Methods for information server selection. *ACM Trans. Inf. Syst.*, 17(1):40–76.

[He and Ounis, 2003] He, B. and Ounis, I. (2003). A study of parameter tuning for term frequency normalization. In *CIKM*, pages 10–16.

[He and Ounis, 2004] He, B. and Ounis, I. (2004). A Query-based Pre-retrieval Model Selection Approach to Information Retrieval. In *Proceedings of RIAO 2004 (Recherche d'Information Assistee par Ordinateur - Computer assisted information retrieval)*, pages 706–719.

[He and Ounis, 2006] He, B. and Ounis, I. (2006). Query performance prediction. *Inf. Syst.*, 31(7):585–594.

[Hiemstra, 2000] Hiemstra, D. (2000). A probabilistic justification for using tf.idf term weighting in information retrieval. *International Journal on Digital Libraries*, 3(2):131–139.

[Hiemstra, 2001] Hiemstra, D. (2001). *Using Language Models for Information Retrieval*. PhD thesis, University of Twente.

[Hiemstra et al., 2004] Hiemstra, D., Robertson, S., and Zaragoza, H. (2004). Parsimonious language models for information retrieval. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 178–185, New York, NY, USA. ACM.

[Jones, 1988] Jones, K. S. (1988). A statistical interpretation of term specificity and its application in retrieval. pages 132–142.

[Jones et al., 2000a] Jones, K. S., Walker, S., and Robertson, S. E. (2000a). A probabilistic model of information retrieval: development and comparative experiments - part 1. *Inf. Process. Manage.*, 36(6):779–808.

[Jones et al., 2000b] Jones, K. S., Walker, S., and Robertson, S. E. (2000b). A probabilistic model of information retrieval: development and comparative experiments - part 2. *Inf. Process. Manage.*, 36(6):809–840.

[Kamps et al., 2004] Kamps, J., de Rijke, M., and Sigurbjörnsson, B. (2004). Length normalization in xml retrieval. In *Proceedings of the 27th annual international ACM SIGIR conference on research and development in information retrieval*, pages 80–87, New York, NY, USA. ACM Press.

[Kamps et al., 2005] Kamps, J., de Rijke, M., and Sigurbjörnsson, B. (2005). The importance of length normalization for xml retrieval. *Inf. Retr.*, 8(4):631–654.

[Kaszkiel and Zobel, 1997] Kaszkiel, M. and Zobel, J. (1997). Passage retrieval revisited. In *Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '97, pages 178–185, New York, NY, USA. ACM.

[Katz, 1996] Katz, S. M. (1996). Distribution of content words and phrases in text and language modelling. *Nat. Lang. Eng.*, 2(1):15–59.

[Kleinberg, 1999] Kleinberg, J. (1999). Authoritative sources in a hyperlinked environment. *Journal of ACM*, 46.

[Kwok, 1996] Kwok, K. L. (1996). A new method of weighting query terms for ad-hoc retrieval. In *SIGIR '96: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 187–195, New York, NY, USA. ACM.

[Kwok, 2005] Kwok, K. L. (2005). An attempt to identify weakest and strongest queries. In *Predicting Query Difficulty - Methods and Applications, ACM SIGIR 2005 Workshop*.

[Lafferty and Zhai, 2003] Lafferty, J. and Zhai, C. (2003). *Probabilistic Relevance Models Based on Document and Query Generation*, chapter 1. Kluwer.

[Lavrenko and Croft, 2001] Lavrenko, V. and Croft, W. B. (2001). Relevance-based language models. In *SIGIR*, pages 120–127.

[Margulis, 1992] Margulis, E. (1992). N-poisson document modelling. In Belkin, N., Ingwersen, P., and Pejtersen, M., editors, *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 177–189, New York.

[Maron and Kuhns, 1960] Maron, M. and Kuhns, J. (1960). On relevance, probabilistic indexing, and information retrieval. *Journal of the ACM*, 7:216–244.

[Mass and Mandelbrod, 2003] Mass, Y. and Mandelbrod, M. (2003). Retrieving the most relevant xml component. In *Proceedings of the Second Workshop of Initiative for The Evaluation of XML Retrieval(INEX)*, pages 53–58, Schloss Dagstuhl, Germanny.

[Ney et al., 1994] Ney, H., Essen, U., and Kneser, R. (1994). On structuring probabilistic dependencies in stochastic language modelling. *Computer Speech and Language*, 8:1–38.

[Ogilvie and Callan, 2003] Ogilvie, P. and Callan, J. (2003). Language models and structured document retrieval.

[Ogilvie and Callan, 2004] Ogilvie, P. and Callan, J. (2004). Hierarchical language models for xml component retrieval. In *INEX*, pages 224–237.

[Ogilvie and Callan, 2005] Ogilvie, P. and Callan, J. (2005). Parameter estimation for a simple hierarchical generative model for xml retrieval. In *INEX*, pages 211–224.

[Ponte and Croft, 1998a] Ponte, J. and Croft, W. (1998a). A language modeling approach to information retrieval. In [Croft et al., 1998], pages 275–281.

[Ponte and Croft, 1998b] Ponte, J. M. and Croft, W. B. (1998b). A language modeling approach to information retrieval. In *SIGIR*, pages 275–281.

[Raghavan and Wong, 1999] Raghavan, V. V. and Wong, S. K. M. (1999). A critical analysis of vector space model for information retrieval. *Journal of the American Society for Information Science*, 37(5):279–287.

[Robertson, 1977] Robertson, S. (1977). The probability ranking principle in IR. *Journal of Documentation*, 33:294–304.

[Robertson, 2004] Robertson, S. (2004). Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation*, 60:503–520.

[Robertson et al., 1994] Robertson, S., S. Walker, S. J., Hancock-Beaulieu, M., and Gatford, M. (1994). Okapi at trec-3. In *Text REtrieval Conference*.

[Robertson and Sparck Jones, 1976] Robertson, S. and Sparck Jones, K. (1976). Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27:129–146.

[Robertson and Walker, 1997] Robertson, S. and Walker, S. (1997). On relevance weights with little relevance information. In Belkin, N., Narasimhalu, D., and Willet, P., editors, *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 16–24, New York. ACM.

[Robertson, 1981] Robertson, S. E. (1981). Term frequency and term value. In *SIGIR*, pages 22–29.

[Robertson et al., 1981] Robertson, S. E., van Rijsbergen, C. J., and Porter, M. F. (1981). Probabilistic models of indexing and searching. In Oddy, R. N., Robertson, S. E., van Rijsbergen, C. J., and Williams, P. W., editors, *Information Retrieval Research*, pages 35–56. Butterworths, London.

[Robertson and Walker, 1994] Robertson, S. E. and Walker, S. (1994). Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In [Croft and van Rijsbergen, 1994], pages 232–241.

[Robertson et al., 1995] Robertson, S. E., Walker, S., and Hancock-Beaulieu, M. (1995). Large test collection experiments on an operational interactive system: Okapi at TREC. *Information Processing and Management*, 31:345–360.

[Robertson et al., 2004] Robertson, S. E., Zaragoza, H., and Taylor, M. J. (2004). Simple BM25 extension to multiple weighted fields. In *CIKM*, pages 42–49.

[Rocchio, 1971] Rocchio, J. (1971). Relevance feedback in information retrieval. In [Salton, 1971].

[Roelleke, 2003a] Roelleke, T. (2003a). A frequency-based and a poisson-based definition of the probability of being informative. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, SIGIR '03, pages 227–234, New York, NY, USA. ACM.

[Roelleke, 2003b] Roelleke, T. (2003b). A frequency-based and a Poisson-based probability of being informative. In *ACM SIGIR*, pages 227–234, Toronto, Canada.

[Roelleke et al., 2002] Roelleke, T., Lalmas, M., Kazai, G., Ruthven, I., and Quicker, S. (2002). The accessibility dimension for structured document retrieval. In *Proceedings of the BCS-IRSG European Conference on Information Retrieval (ECIR), Glasgow*.

[Roelleke and Wang, 2006] Roelleke, T. and Wang, J. (2006). A parallel derivation of probabilistic information retrieval models. In *ACM SIGIR*, pages 107–114, Seattle, USA.

[Roelleke and Wang, 2007] Roelleke, T. and Wang, J. (2007). Probabilistic logical modelling of the binary independence retrieval model. In *Proceedings of the 1st International Conference on Theory of Information Retrieval (ICTIR 07) - Studies in Theory of Information Retrieval*.

[Roelleke et al., 2008] Roelleke, T., Wu, H., Wang, J., and Azzam, H. (2008). Modelling retrieval models in a probabilistic relational algebra with a new operator: The relational Bayes. *VLDB Journal*, 17(1):5–37.

[Salton, 1971] Salton, G., editor (1971). *The SMART Retrieval System - Experiments in Automatic Document Processing*. Prentice Hall, Englewood, Cliffs, New Jersey.

[Salton and Buckley, 1988] Salton, G. and Buckley, C. (1988). Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523.

[Salton et al., 1983] Salton, G., Fox, E., and Wu, H. (1983). Extended Boolean information retrieval. *Communications of the ACM*, 26:1022–1036.

[Salton et al., 1975] Salton, G., Wong, A., and Yang, C. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18:613–620.

[Si et al., 2002] Si, L., Jin, R., Callan, J., and Ogilvie, P. (2002). A language modeling framework for resource selection and results merging. In *In CIKM 2002*, pages 391–397. ACM Press.

[Singhal et al., 1996] Singhal, A., Buckley, C., and Mitra, M. (1996). Pivoted document length normalistation. In Frei, H., Harmann, D., Schäuble, P., and Wilkinson, R., editors, *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–39, New York. ACM.

[Thomas M. Cover, 2006] Thomas M. Cover, J. A. T., editor (2006). *Elements of Information Theory*. John Wiley & Sons, Inc.

[Turtle and Croft, 1991] Turtle, H. and Croft, W. (1991). Efficient probabilistic inference for text retrieval. In *Proceedings RIAO 91*, pages 644–661, Paris, France.

[Turtle and Croft, 1990] Turtle, H. and Croft, W. B. (1990). Inference networks for document retrieval. In Vidick, J.-L., editor, *Proceedings of the 13th International Conference on Research and Development in Information Retrieval*, pages 1–24, New York. ACM.

[van Rijsbergen, 1986] van Rijsbergen, C. J. (1986). A non-classical logic for information retrieval. *The Computer Journal*, 29(6):481–485.

[van Rijsbergen, 1989] van Rijsbergen, C. J. (1989). Towards an information logic. In Belkin, N. and van Rijsbergen, C. J., editors, *Proceedings of the Twelfth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 77–86, New York.

[Wang and Roelleke, 2006] Wang, J. and Roelleke, T. (2006). Context-specific frequencies and discriminativeness for the retrieval of structured documents. In *European Conference on Information Retrieval (ECIR), London*. Poster.

[Wong and Yao, 1995] Wong, S. and Yao, Y. (1995). On modeling information retrieval with probabilistic inference. *ACM Transactions on Information Systems*, 13(1):38–68.

[Wong et al., 1985] Wong, S., Ziarko, W., and Wong, P. (1985). Generalized vector space model in information retrieval. In *Proceedings of the 8th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 18–25, New York. ACM.

[Wong and Raghavan, 1984] Wong, S. K. M. and Raghavan, V. V. (1984). Vector space model of information retrieval: a reevaluation. In *SIGIR '84: Proceedings of the 7th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 167–185, Swinton, UK, UK. British Computer Society.

[Wu and Salton, 1981] Wu, H. and Salton, G. (1981). A comparison of search term weighting: term relevance vs. inverse document frequency. *SIGIR Forum*, 16(1):30–39.

[Yu et al., 1982] Yu, C., Lam, K., and Salton, G. (1982). Term weighting in information retrieval using the term precision model. *Journal of the ACM*, 29(1):152–170.

[Yuwono and Lee, 1997] Yuwono, B. and Lee, D. L. (1997). Server ranking for distributed text retrieval systems on the internet. In *In Proceedings of the Fifth International Conference on Database Systems for Advanced Applications*, pages 41–49.

[Zaragoza et al., 2003] Zaragoza, H., Hiemstra, D., and Tipping, M. (2003). Bayesian extension to the language model for ad hoc information retrieval. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 4–9, New York, NY, USA. ACM Press.

[Zhai and Lafferty, 2004] Zhai, C. and Lafferty, J. (2004). A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214.

[Zhai and Lafferty, 2002] Zhai, C. and Lafferty, J. D. (2002). Two-stage language models for information retrieval. In *ACM SIGIR*, pages 49–56.

Index